AD/A-002 825

VEHICLE DETECTION IN EMPLACED SENSOR FIELDS: A USER'S GUIDE TO A SIMULATION MODEL AND A TRACK-IDENTIFICATION ALGORITHM

Morton B. Berman

RAND Corporation

Prepared for:

Deputy Chief of Staff, Research and Development (Air Force)

January 1973

AD A002825

014136

R-1186-PR

January 1973

Vehicle Detection in Emplaced Sensor Fields: A User's Guide to a Simulation Model and a Track-Identification Algorithm

Morton B. Berman

A Report prepared for

UNITED STATES AIR FORCE PROJECT RAND

25th Year
Rand
SANTA MONICA, CA. 90406

DOCUMENT CONTROL DA[TA]

AD/A-002825

1. ORIGINATING ACTIVITY

The Rand Corporation

2a. REPORT SECURITY CLASSIFICATION

UNCLASSIFIED

2b. GROUP

3. REPORT TITLE

VEHICLE DETECTION IN EMPLACED SENSOR FIELDS: A USER'S GUIDE TO A SIMULATION MODEL AND A TRACK-IDENTIFICATION ALGORITHM

4. AUTHOR(S) (Last name, first name, initial)

Berman, Morton B.

5. REPORT DATE

January 1973

6a. TOTAL NO. OF PAGES

111

6b. NO. OF REFS.

7. CONTRACT OR GRANT NO.

F44620-73-C-0011

8. ORIGINATOR'S REPORT NO.

R-1186-PR

9a. AVAILABILITY/LIMITATION NOTICES

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

9b. SPONSORING AGENCY

United States Air Force
Project RAND

10. ABSTRACT

Documentation and listing of two computer programs intended to provide the Air Force with means of exploring the effectiveness of various detection devices for purposes of interdicting vehicles. The analytical basis for the programs is given in R-1187, which should be used in conjunction with this report. Each program is separately documented, giving all the information needed to operate it. The simulation model simulates detection patterns of vehicles passing through fields of magnetic, acoustic, or seismic sensors, under varying vehicle flow and background (false alarm) conditions. It also provides inputs to the pattern recognition algorithm allowing precise verification of the pattern recognizers' detection ability. The algorithm also accepts real world data, and is adaptive in eliminating from consideration sensors that have provided incorrect information in the past. Together, the programs are a first step toward automating the process of evaluating the sensor information.

11. KEY WORDS

PATTERN RECOGNITION
SENSORS
COMPUTER SIMULATION
SURVEILLANCE

R-1186-PR
January 1973

Vehicle Detection in Emplaced Sensor Fields: A User's Guide to a Simulation Model and a Track-Identification Algorithm

Morton B. Berman

A Report prepared for

UNITED STATES AIR FORCE PROJECT RAND

Rand
SANTA MONICA, CA 90406

PREFACE

The two computer programs described in this report—a simulation model of vehicle traffic through fields of emplaced sensors, and a pattern detection algorithm of two-way traffic through an emplaced sensor field—were developed as part of Rand's investigation of the employment of USAF tactical air forces for interdiction campaigns. The study was in part responsive to a request by the Commander, Eglin Air Force Base, for assistance in conducting and evaluating the Dune Moon sensor-system tests initiated by former Secretary of Defense Robert McNamara.

The programs described here will provide the Air Force with a means for exploring the effectiveness of various types of emplaced sensor fields. In addition, they permit a first step toward creating an automatic capability that should prove superior to the manual methods currently used to evaluate information provided by fields of emplaced sensors.

Full information to operate both computer programs is provided. The report is intended to assist Air Force and other Department of Defense agencies in exploring the usefulness of emplaced sensor fields, in making decisions concerning their use, and in designing and operating appropriate data-processing techniques. This report should be used in conjunction with Anthony P. Ciervo, *Automatic Track Identification: An Adaptive Pattern Recognition Algorithm*, The Rand Corporation, R-1187-PR, January 1973

SUMMARY

This report provides the necessary information for using two separate computer programs: (1) a simulation model of vehicle traffic through fields of emplaced magnetic, acoustic, or seismic sensors; and (2) a pattern detection algorithm of two-way vehicle traffic through a field of emplaced sensors.

The simulation model supplies the user with a device to simulate detection patterns of different sensors under varying vehicle-flow and background (false-alarm) conditions. In addition, it provides inputs to the pattern detection algorithm allowing precise verification of the pattern recognizer's ability to detect vehicle flow.

The pattern detection algorithm is designed to accept inputs from the simulation model to allow proper selection of critical parameters. It is also designed to accept real-world data--both actual detections and false alarms--of vehicle flow past fields of sensors. The algorithm is adaptive in that it will eliminate from consideration sensors that provide incorrect information based on previous performance. The pattern detection algorithm, when used with the simulation, will specify the number of vehicle convoys actually detected and the number of vehicle convoys incorrectly identified. Actual times of detection and direction are produced for each supposed convoy identified.

This report should be used in conjunction with Rand report R-1187-PR (see Preface above), which describes the analytical basis for these two computer programs.

CONTENTS

PART A: THE SIMULATION MODEL

I. INTRODUCTION

The simulation model described here deals with the detection of vehicle flow through a field of emplaced sensors. The model permits the user to explore the sensitivity of various sensors to single or multiple vehicle traffic in two directions, simultaneously, through a string of sensors.* Three levels of background disturbance (false alarms) may be specified by the user. The individual sensor detections--vehicles and false alarms--may be stored on an external medium (tape or disk) for later use by the pattern recognition algorithm (described in Part B), thus serving as a verification and parameter-selection device for that algorithm.

The simulation model was written in the SIMSCRIPT II† language for use on The Rand Corporation IBM 360/65 computer. It requires a region size of 76 k bytes and standard input and output devices. Written as an experimental model for that machine, it may require some modifications for other installations.

This part of the report provides a brief description of the processing performed by the simulation model and controlled by the user, some limitations of the model, a detailed description of inputting data to the model, a description of simulation output, and an interpretation of error messages provided by the simulation.

---

*An analytical treatment of this model is described in R-1187-PR, cited in the Preface.

†The particular version was SIMSCRIPT II.5 marketed by Consolidated Analysis Centers, Inc.

Preceding page blank

II. PROCESSING PERFORMED

Single or multiple vehicle traffic (convoys) traversing a sensor string in two directions may be simulated with the model. Sensor sensitivity may be specified or based on randomly drawn degradation from some nominal sensitivity, depending on the actual sensor characteristics the user wishes to duplicate. Each sensor is capable of multiple detections as long as a vehicle is within the sensor's sphere of influence and the sensor is capable of detection. Detection would not be possible if the sensor were in the dead-time period immediately following a previous detection.

Realism is introduced into the simulation by permitting sensors to be exposed to several levels of background disturbances (false alarms) that can also result in detections. The output of the model includes statistics on convoys, convoy size, and average detections for each sensor per convoy, per unit time, and per vehicle. Statistics are also presented on false alarms. The model is capable of a visual display of detection patterns and storage (on tape or disk) of detections for use by the pattern detection algorithm.

The user may specify the following:

1. The length of the road segment along which a specified number of sensors are to be equidistantly emplaced.
2. The probability of detection of a vehicle within a sensor's sphere of influence. This is expressed as an isosceles triangular distribution (except in the case of magnetic sensors, which can be specified to have a single detection probability p). The accuracy of the detection computation is controlled by an integration step size specified by the user; the smaller the integration step size (i.e., increasing the number of divisions of each triangle), the more accurate the determination of a detection of a vehicle or group of vehicles. However, too large a number of integration increments severely slows simulation execution. We have experienced good results

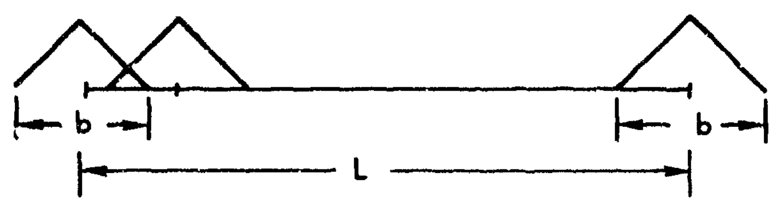
for 1 to 9 vehicle convoys and velocities of 10 to 50 mph using 50 to 75 increments.

3. The base of each triangle, which would represent the sphere of influence of a perfect sensor emplaced in the center of the road segment.
4. A normal distribution of location deviations a sensor would experience if emplaced by air. The simulation can thus introduce the realism of air-delivered sensors by degrading the sphere of influence of each sensor. Further, realism is introduced by allowing the user to specify a probability that the sensor will be destroyed on impact. If the user desires, all triangles representing sensors may have identical characteristics.
5. The time a sensor cannot transmit immediately after a detection.
6. The spacing between vehicles in a convoy as a function of convoy velocity.
7. Convoy generation rate as a Poisson process, the probability of various numbers of vehicles in a convoy, and beta distribution of convoy velocities, which change at each sensor.
8. Three levels of background disturbances (false alarms): ambient, low, and high. These would represent the environment the user wishes to simulate. Each false-alarm level occurs as a Poisson process, with a uniform random duration specified for each. If two false-alarm levels arrive simultaneously, the one with the higher level is considered active. All sensors are exposed equally to false alarms, which are also a Poisson process.
9. Whether a visual-time printout of actual detections is desired. This printout shows vehicle detections (differentiated by direction), false-alarm detections, and first and last vehicle passage past the midpoint of each sensor.
10. The total number of convoys to be simulated or the total time to be simulated.

III. RESTRICTIONS AND LIMITATIONS

The following restrictions and limitations of the model should be observed:

1. Each sensor is equidistantly spaced along the road segment, L, as illustrated in the diagram below.



Thus, detections may start as early as b/2 before the road segment; likewise, they may end as late as L + (b/2).

2. Because of computational restrictions, sensor midpoints may not be closer to one another than b/2.
3. Although the simulation accommodates traffic flowing in two directions, it does not permit vehicles passing one another in one direction. Whenever a convoy enters a sensor's sphere of influence and there is already a convoy of slower velocity ahead of it, the velocity of the convoy will be set equal to that of the earlier convoy.
4. Computations of detections of convoys going in opposite directions through the same sensor are performed independently.
5. There must be a minimum of three sensors in a string.

IV. HOW TO USE THE SIMULATION MODEL

INPUT DATA FORMATS

A datum requiring a decimal point is shown by (D); a datum requiring an integer value is shown by (I); and a character datum is shown by (A). Other parenthetic symbols represent the variable name internal to the program.

Data are entered in free format. The only restrictions are that there must be at least one space between entries, and a datum of zero must be entered. Data may be continued on a new card.

New Card--Comments Card

Any comments, taking up to eighty columns, the user desires to title the output listing.

New Card(s)--Sensor and Control Information

a. Print control (PRINT.MAP) (I)

If a graph of detections and false alarms is desired as the simulation progresses, enter 1. Otherwise, enter 0.

b. Number of convoys (MAX.NBR.CONVS) (I)

The maximum number of complete convoy passages desired before stopping the simulation.

c. Time control (MAX.TIME) (D)

The maximum length of simulated time (minutes) allowed for the simulation. Simulation halts at the first completed convoy passage whose completion time is greater than this value.

d. Road segment length (ROAD.LGNTH) (D)

Length of road segment in meters.

e. Number of sensors (N.SENSORS) (I)

Number of equally spaced sensors along the road segment. Must be greater than 2.

f. Integration increments (I)

Enter 1, if magnetic sensors are desired; otherwise, enter the number of triangular distribution increments.

g. Maximum permitted triangle base (NOMINAL.BASE) (D)

b, base length, in meters, of a perfect sensor dropped in the center of the road.

h. Destruction probability (PROB.DEAD) (D)

The probability ($0 < p < 1$) a sensor will be destroyed on impact.

i. Normal distribution standard deviation (STD.DEV) (D)

The standard deviation in meters of the specified normally distributed drop pattern of sensors about the road.

j. Sensor dead time (DEAD.TIME) (D)

The time, in minutes, that the sensor cannot transmit after a detection. Must be greater than zero.

k. Coefficient (AREA) (D)

(1) If the sensors to be simulated are magnetic sensors, enter the probability of the sensors' ability to detect. Must be less than or equal to 1.0.

(2) For runs where a constant base is desired for all sensors, enter the height of the sensor triangle. Each base will be the value entered for maximum base in item g above.

(3) For simulation of sensors distributed about the road, enter the coefficient c_1.*

l. Coefficient two (C.2) (D)

For simulation of sensors distributed about the road, enter the coefficient c_2.* Otherwise, enter 0.0.

---

*See R-1187-PR for a discussion of how these constants should be computed.

m. Random seed (SEED.V(8)) (I)

Enter any of the ten random seeds shown in Appendix A. The sequence of random numbers thus started selects the distribution of sensors about the road. On subsequent runs, entering the same number will yield the same random number stream, or entering a new number will yield a different stream.

n. Base control flag (RAN.NBR) (D)

If all sensor triangles are to have the same base, enter 1.0. The base will be that in item g above. Then enter the height of the triangles in item k (2) above. Otherwise, enter 0.0.

o. Truck spacing (SPACE.FACTOR) (D)

Enter the spacing, in meters, desired between trucks for each kilometer per hour of convoy velocity.

p. Constant convoy generation flag (CONSTANT.CON) (I)

Entering 1 will cause all convoys to be generated with a constant interarrival time at the specified mean rate. Otherwise, enter 0.0.

New Card(s)--Trucks in Convoy Distribution

<u>Pairs of Values</u>

1st Cumulative prob, p_1 (D) of number of trucks, T_1 (I)
2d Cumulative prob, p_2 (D) of number of trucks, T_2 (I)
⋮ ⋮ ⋮ ⋮
nth Cumulative prob, p_n (D) of number of trucks, T_n (I)
Enter an asterisk (*) after T_n.

New Card(s)--Westbound Convoy Information

a. Convoy rate (RATE.CNV) (D)

Mean of a Poisson distribution. The average number of convoys generated per minute.

b. Mean convoy velocity (MEAN.VEL) (D)

Mean of a beta distribution. The average velocity of the convoys in kilometers per hour.

c. Modal convoy velocity (MODE.VEL) (D)

Mode of a beta distribution. The modal velocity of the convoys in kilometers per hour.

d. Upper bound on velocity (UPPER.BND) (D)

The highest possible velocity of a convoy in kilometers per hour.

e. Lower bound on velocity (LOWER.BND) (D)

The lowest possible velocity of a convoy in kilometers per hour. Must be greater than zero.

f. Direction name (DIR. SYMBOL) (A)

The convoy direction may be designated by using any four letters. For convenience, WEST is suggested. Such a convoy will always start at sensor 1.

New Card(s)--Eastbound Convoy Information

Enter items a through e as for the westbound convoy. The values may be different. The entry for f can be EAST or any other four-letter symbol. Convoys for this direction always start at the nth sensor.

New Card(s)--False-Alarm Information

a. Ambient arrival rate (LAM) (D)

This is the arrival rate of false alarms of the lowest density (level 1). It is the mean of a Poisson process. It is entered as number per minute for a single sensor.

b. Medium arrival rate (LAM) (D)

Arrival rate of false alarms of the medium density (level 2); false alarms in number per minute for a single sensor.

c. Maximum-level, 2-burst length (UB) (D)

Upper bound of a uniform distribution determining burst length of a level-2 false alarm in minutes.

d. Minimum-level, 2-burst length (LB) (D)

Lower bound of the uniform distribution that determines burst length of a level-2 false alarm in minutes.

e. Arrival rate of level-2 bursts (GAM) (D)

The rate at which bursts of level 2 arrive. The mean of a Poisson process in number per minute.

f. High arrival rate (LAM) (D)

Arrival rate of false alarms of the highest density (level 3); false alarms in number per minute for a single sensor.

g. Maximum-lovel, 3-burst length (UB) (D)

Upper bound of a uniform distribution that determines burst length of a level-3 false alarm in minutes.

h. Minimum-level, 3-burst length (LB) (D)

Lower bound of the uniform distribution that determines burst length of a level-3 false alarm in minutes.

i. Arrival rate of level-3 bursts (GAM) (D)

The rate at which bursts of level 3 arrive. The mean of a Poisson process in number per minute.

A SAMPLE DATA DECK

The following figure (from Appendix B) shows a sample data deck punched from the above input data formats:

```
   THIS IS AN EXAMPLE OF THE SIMULATION MODEL  2 DEC 1972
 1  20  1000.0  1000.0  5  50  250.0  0.2  20.0  0.15  40.0  1000.0 8108509
0.0  1.0  0
    0.1  1   0.3  2   0.4  3   0.6  4   0.9  5   1.0  6  *
        0.1  24.0  20.0   35.0   15.0  WEST
        0.06 24.0  20.0   35.0   15.5  EAST
0.5  1.5   6.0  0.5  0.2   3.0  0.5  0.01  0.05
```

A 1000-meter road segment has been selected on which five sensors have been equidistantly spaced and dropped randomly. Vehicles traverse the road segment in two directions.

INTERPRETING THE SIMULATION OUTPUT

Figure 1 shows a sample of the description of the input data deck and some of its ramifications. This description is produced by each simulation run. The first line after the heading is a repeat of the comment card in the data deck. The second line gives the data controlling the length of the simulation in convoys generated and/or total time.

```
                 ***** SIMULATION OF TRUCK CONVOYS MOVING IN TWO DIRECTIONS THRU A SENSOR FIELD *****

   THIS IS AN EXAMPLE OF THE SIMULATION MODEL  2 DEC 1972
SIMULATION HALTS IF TIME EXCEEDS 1000.000 MIN. OR CONVOYS GENERATED EXCEEDS   20

                    *** SENSOR PARAMETERS ***

ROUTE SEGMENT IS   1000.00 M. WITH   5 EQUALLY SPACED SENSORS.
EACH WITH NOMINAL BASE OF  250.00 M.
THE COEFF. FOR COMPUTING HEIGHT ARE (C1)       40.00 (C2)    1000.00
EVERY SENSOR TRIANGLE HAS   30 INCREMENTS.
THE PROB. THE SENSOR IS DEAD ON IMPACT IS  .200. STANDARD DEVIATION
FROM THE ROAD IS  20.0. THE DEAD TIME OF A SENSOR AFTER ACTIVATION
IS  .150 MIN. THE RANDOM SEED FOR SELECTING TRIANGLES IS 8108509.0


                    *** SENSOR ATTRIBUTES ***
         SENSOR   BASE(M.)  SLOPE X DELTAS   DELTAS(M.) AREA  DEAD(=1)
            1      250.00         .00160       5.000    5.00     0
            2      247.73         .00125       4.955    3.87     0
            3      248.05         .00129       4.961    3.99     0
            4        0.         0.             0.       3.99     1
            5      249.00         .00144       4.982    4.47     0

                    *** CONVOY INFORMATION ***

DISTANCE BETWEEN TRUCKS IN A CONVOY IS   1.0000 M. FOR EACH KM/HR.
          DISTRIBUTION OF TRUCKS IN A CONVOY
              TRUCKS    CUMULATIVE PROB
                 1          .100
                 2          .300
                 3          .400
                 4          .600
                 5          .900
                 6         1.000


THE WEST DIRECTION HAS A CONVOY RATE OF      .100 PER MIN. WHICH IS ONE
CONVOY EVERY     10.000 MIN.  THE AVG VELOCITY IS    24.0 KM/HR.
THE MODE VELOCITY IS   20.00 KM/HR.        THE SLOWEST IS   15.00 KM/HR.
AND THE FASTEST IS    35.00 KM/HR..  K1 IS  2.125 AND K2 IS  2.375


THE EAST DIRECTION HAS A CONVOY RATE OF      .060 PER MIN. WHICH IS ONE
CONVOY EVERY     16.667 MIN.  THE AVG VELOCITY IS    24.0 KM/HR.
THE MODE VELOCITY IS   20.00 KM/HR.        THE SLOWEST IS   15.50 KM/HR.
AND THE FASTEST IS    35.00 KM/HR..  K1 IS  2.144 AND K2 IS  2.401

                    *** FALSE ALARM INFORMATION ***
ALARM LEVEL   ALARMS/MIN  MIN BURST (MIN) MAX BURST (MIN)  ARR/MIN
     1           2.500
     2           7.500        .500            6.000          .200
     3          15.000        .010             .500          .050
```

Fig. 1 — Description of input data deck

Under SENSOR PARAMETERS are the length of the road segment in meters and the number of sensors. The second line shows the base size of a perfect sensor emplaced on the road segment, followed by the two coefficients used to determine the height of the triangular distribution of the represented sensor. The next line indicates the number of increments each triangle will have for computing the probability of detections. The next three lines contain several items: the user-inputted probability that a sensor will be destroyed on impact, the standard deviation (meters) of the normal distribution that is used for randomly selecting the distance from the road that a sensor is emplaced after being dropped, the dead time of the sensor, and the random seed. The random seeds determine the starting point of random draws from the normal distribution.

SENSOR ATTRIBUTES are the resulting characteristics of each sensor after its distance from the center of the road is randomly selected from the normal distribution. The effective base is shown for each sensor. A zero indicates the sensor was destroyed on impact or landed too far from the road to be effective. If it was destroyed on impact (shown by a 1 in the column labeled DEAD), it will never transmit false alarms. The column labeled SLOPE X DELTAS gives the factor that, when multiplied by the position of a vehicle in the sensor field of influence, will yield the probability of detection at that point. The column labeled DELTAS(M.) gives the width of each increment of the triangular distribution. During the simulation, a vehicle moves this increment before a calculation of the probability of detection is made. The column labeled AREA is the area of each sensor's triangular distribution and represents the sensitivity of the sensor for detecting vehicles of a given velocity.

Under CONVOY INFORMATION the distance between trucks in a convoy comes directly from the input data, as does the size of convoy distribution. Each time a convoy is generated, a uniform random variate is drawn to determine the number of trucks in the convoy. For example, if a number between 0.6 and 0.9 is drawn, the convoy will have five trucks, or if a number between 0.0 and 0.1 is drawn, it will have one truck.

The next two paragraphs in Fig. 2 reflect the information on convoys traveling in the west and east directions and start with the mean rate (Poisson process) at which convoys will be generated, followed by the beta distribution information on convoy velocity. K1 and K2 are the parameters of the distribution computed from this information. Each time a convoy starts through a sensor, its velocity is computed by random sampling from this distribution.

Under FALSE-ALARM INFORMATION, the three alarm levels are shown (1 = ambient, 2 = low or medium, 3 = high). They *must* have increasing ALARMS/MIN, which is the mean of the Poisson process that generates false alarms to sensors when a particular level is on. The ARR/MIN column gives the mean arrival rate (Poisson process) of each level of false alarms. Of course, if neither alarm-level 2 nor 3 is on, level 1 (ambient) will be. MIN BURST (MIN) and MAX BURST (MIN) are the inputted minimum and maximum times (minutes) of the uniform distribution of the duration of level-2 and level-3 false alarms.

Figure 2 shows an example of the printed output the user can obtain by exercising the print-control option (field "a" of the sensor- and control-input data). The simulated time in minutes is shown down the left side. The increment of time is the sensor dead time. In the column for each sensor, E or W represents a vehicle detection for a convoy traveling from the east or west. The + or - represents the passage of the first and last trucks of the convoy past the midpoint of a sensor (+ is for convoys from the east, and - for convoys from the west). Usually these signs are paired, but occasionally only one is shown. A single occurrence is the result of (1) the passage of only a single vehicle or (2) a convoy speed that is too fast relative to the time increments printed. The symbols 1, 2, * represent false alarms of level 1, 2, and 3, respectively.

An example of summary statistics relating to false alarms is given in Fig. 3. SYSTEM FALSE ALARMS are the number of times each alarm level was on during the simulation. This item is shown under NO. OF BURSTS. The average length of time each was on is shown under AVG. BURST LENGTH (MIN). The number of times each level was on divided by total simulated time is shown under AVG BURST ARR/MIN. For each sensor the average

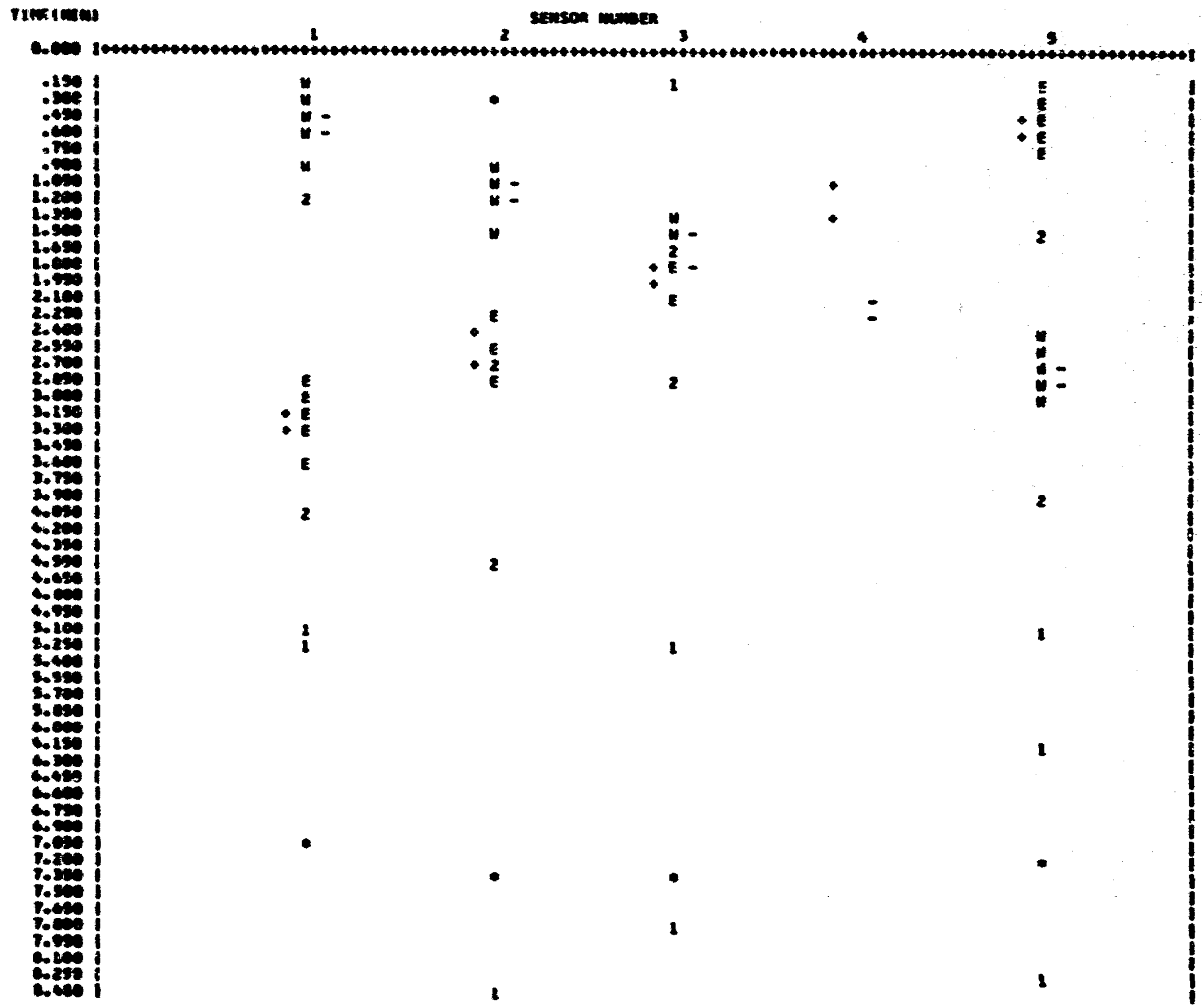


Fig. 2 — Printed output option

**** SYSTEM FALSE ALARMS ****

ALARM LEVEL	NO. OF BURSTS	AVG. BURST LENGTH(MIN)	AVG BURST ARR/MIN
1	16	4.3309	.1291
2	15	3.8771	.1107
3	5	.2705	.0515

**** SENSOR FALSE ALARMS ****

SENSOR	ALARMS/MIN : LEVEL 1	LEVEL 2	LEVEL 3	ALL LEVELS
1	.245	.482	.032	.759
2	.182	.427	.032	.640
3	.261	.419	.024	.703
4	0.	0.	0.	0.
5	.253	.490	.024	.767

Fig.3 — False-alarm summary statistics

number of false alarms per minute activating the sensor is shown for each alarm level and the sum of all alarm levels. Notice the sensor that was destroyed on impact does not receive any activations.

Figure 4 is an example of summary statistics relating to the number and type of convoys generated during the simulation. The summary information is presented for each direction and contains the frequency of each convoy size generated and the total number of trucks generated from that convoy size. Relevant totals are also shown.

**** NUMBER OF CONVOYS GENERATED ****

DIRECTION : WEST

CONVOY SIZE	FREQUENCY	TOTAL TRUCKS GENERATED
1	2	2
2	3	6
3	2	6
4	1	4
5	1	5
6	1	6
TOTALS:	10	29

DIRECTION : EAST

CONVOY SIZE	FREQUENCY	TOTAL TRUCKS GENERATED
1	0	0
2	3	6
3	0	0
4	1	4
5	4	20
6	2	12
TOTALS:	10	42

GRAND TOTALS:	20	71

Fig.4 — Convoy summary statistics

Figure 5 shows summary statistics on sensor activations resulting from vehicles. For each sensor, the average number of detections over the simulation is shown under DETECTIONS/MINUTE; the average number of detections per generated convoy is shown under DETECTIONS/CONVOY; and

the average number of detections over all generated trucks is shown under DETECTIONS/TRUCK.

**** AVERAGE CONVOY DETECTIONS BY SENSOR ****

SENSOR	DETECTIONS/MINUTE	DETECTIONS/CONVOY	DETECTIONS/TRUCK
1	.5454	3.4500	.9718
2	.4663	2.9500	.8310
3	.4980	3.1900	.8873
4	0.	0.	0.
5	.5296	3.3500	.9437

Fig.5 — Summary of sensor activations caused by vehicles

INTERPRETING ERROR MESSAGES

The simulation makes basic checks on the input data for compatibility. If some of the input rules have been violated, error messages of the following type appear after the erroneous data is read:

??? ERROR IN ABOVE LINE.

The processing then halts. The user thus has the incorrect value and an indication that it is incorrect. He should examine the data input to see which rule was violated. Examples of violations are negative road length, a lower alarm rate for level 3 than level 2, and the probability of destroying a sensor on impact greater than 1.0.

THE OUTPUT FILE

An output file, in the proper format for use by the detection algorithm, is produced as normal output of the simulation. It contains all actual detections (as well as those caused by false alarms), convoy size, and entry/exit times of convoys into/out of the sensor string. See Appendix D2 for the format of the output file.

ADDITIONAL INFORMATION

Appendix B contains an example of a fully setup data deck for execution of a simulation on the Rand IBM 360/65 computer installation. Appendix C is a full source listing of the entire program.

PART B: THE PATTERN DETECTION ALGORITHM

I. INTRODUCTION

The pattern detection algorithm* presented here will identify vehicle tracks in two directions through a field of emplaced sensors. The algorithm is an adaptive detection mechanism--it continually measures the performance of each sensor in the string, giving more weight to information from reliable sensors and less to unreliable sensors. When a sensor is deemed very unreliable by a user-inputted criterion, it is dropped from consideration entirely.

The algorithm is designed to operate from data provided by the simulation model described in Part A or from real-world data. The algorithm is programmed in FORTRAN IV for the Rand IBM 360/65 installation and requires 178 k bytes of core.†

Part B of this report provides a brief description of the processing performed by the algorithm, some limitations of the algorithm, a detailed description of how to input data to the model, a description of typical algorithm output, an interpretation of error messages provided by the program, and the format required for inputting actual sensor detection data.

---

*Detailed analytical description of the algorithm is provided in R-1187-PR.

†Use of this program at other installations that do not have graphic capability would require removal of the graphic routines and other minor modifications.

II. PROCESSING PERFORMED

The algorithm begins by examining clusters of activations on a sensor. If there are at least ω activations no more than β minutes apart, the cluster is defined as a *valid strip* and a window is opened on the adjacent sensors (see Fig. 6, p. 25). The window length is primarily a function of anticipated vehicle velocity. Thus a conjectured vehicle track (trajectory) is initiated. A valid strip intersecting a window is defined as an *admissible strip*. If a conjectured vehicle track contains at least M admissible strips, the trajectory is confirmed as a vehicle track. Each admissible strip in a confirmed vehicle track is called an ASTI (admissible strip contributing to a track identification). The adaptive logic of the algorithm uses the information on valid strips, admissible strips, and ASTIs for each sensor over time periods of length B to determine the sensor's reliability. High-reliability sensors are given more weight and low-reliability sensors less weight in confirming vehicle tracks. A sensor that continues to have low reliability is eventually eliminated from the string. Thus, on the basis of prior information, the algorithm is capable of adapting its detection ability.

When used in conjunction with the simulation model output, the algorithm will provide statistics on the actual number of convoys detected and missed. The size of convoy, the weights of each sensor for each update, and the number of vehicle tracks identified that were not caused by vehicles will be determined. Graphic output also allows the user to observe the step-by-step process of identification and the actual time a particular track is confirmed.

The user may specify the following:*

1. The anticipated average, minimum, and maximum velocities of convoys traversing the sensor string in either direction.
2. β, the maximum interval between activations in a valid strip.

---

*See R-1187-PR for precise definition of terms and analytical treatment of the algorithm.

3. ω, the minimum number of activations in a valid strip.
4. M, the number of admissible strips required for a trajectory confirmation.
5. The length of the road segment.
6. The number of sensors in the string and their position along the string.
7. B, the number of trajectories (track identifications) to be confirmed before updating sensor performance information (weights).
8. ρ, the back weight for the smoothing function.
9. C, weight below which an inferior sensor is removed from the string after D consecutive weight updates.
10. W, a lower bound on sensor weights for confirming vehicle tracks. The sum of the sensor weights for those sensors contributing admissible strips must be greater than W to confirm a trajectory.
11. A control parameter allowing visual graphs of sensor activity, window activity, track-identification times, false alarms, and vehicle activations.

III. RESTRICTIONS AND LIMITATIONS

As currently coded, the following maximums must be adhered to:

1. B, the number of vehicle tracks to be confirmed (maximum of 50) before updating sensor performance information (weights).
2. Maximum of 15 sensors. (Can be increased to 50 by redimensioning all common arrays in the program. Sufficient information is provided in the MAIN routine for a programmer to make the change.)
3. Maximum of 20 trucks per convoy.

As might be expected, a large number of patterns are possible from various combinations of valid and admissible strips. Every attempt has been made to anticipate anomalous behavior of various trajectory paths.

Figure 6a shows what might be regarded as the most typical behavior and is to be expected in the majority of vehicle tracks. A valid strip on the first (or a subsequent) sensor opens a window on the second sensor in which a valid strip falls, thus opening a window on the third sensor, and so on. Finally, the vehicle track is confirmed.

Figure 6b presents a case in which a valid strip fails to fall in a window. The window is just extended to the average convoy velocity defined by the user. (This illustration requires three admissible strips to confirm a vehicle track.)

Figure 6c shows that a vehicle track once begun will be continued until the end of the string. Although this may seem unnecessary, since a vehicle track with only one admissible strip can never be confirmed, it results in faster computation time and causes no problems.

Figure 6d illustrates a condition that occurs occasionally. At the second sensor two valid strips fall in the same window, and their windows on the subsequent sensor overlap. In this case the program considers the extremes as one large window.

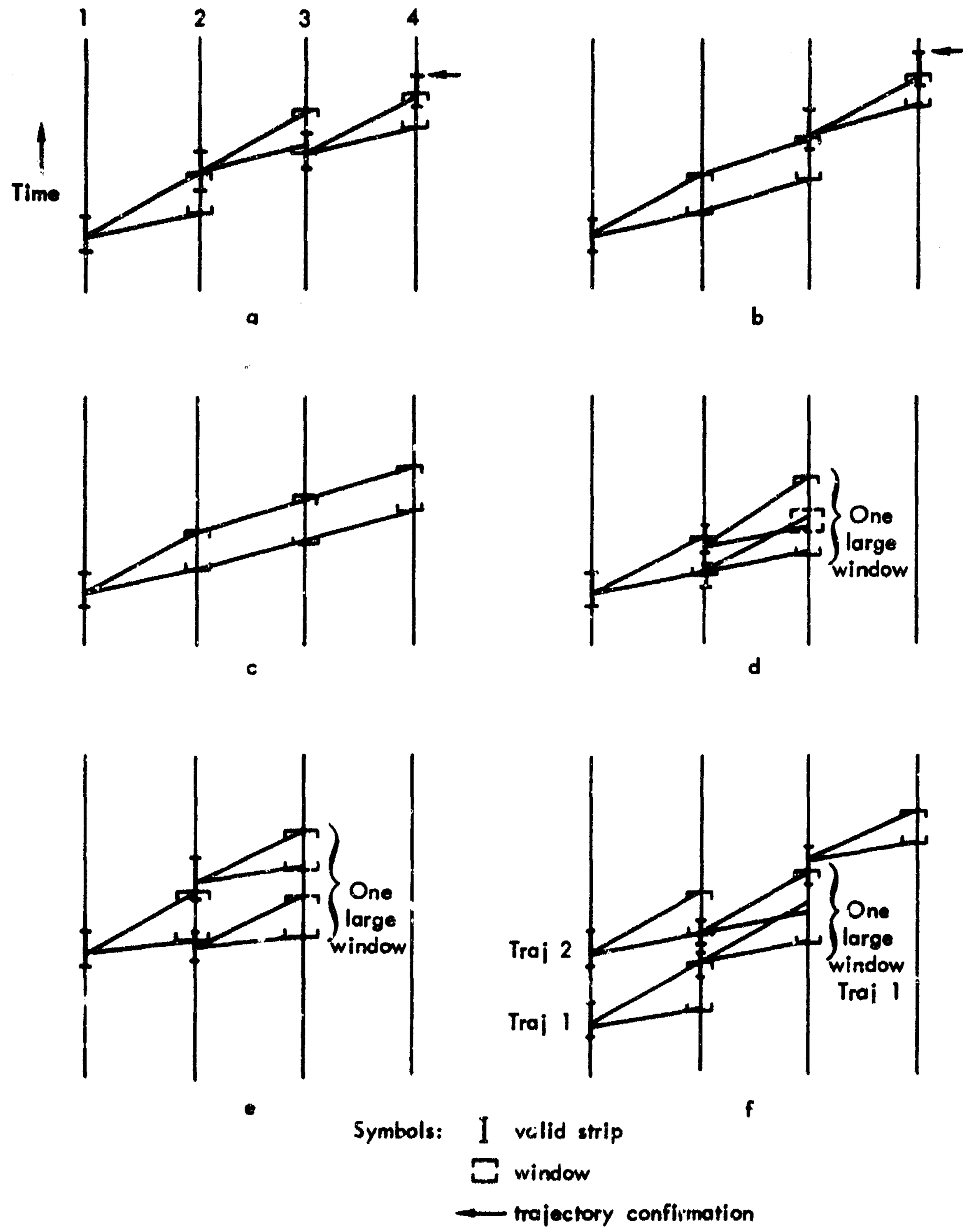


Fig.6 — Illustration of program's handling of anomalous behavior

Figure 6e shows a similar condition, but the windows on the subsequent sensor do not overlap. The extremes are still treated as one large window--*except* if the lower window was closed before the upper window was opened. (Windows are scanned for possibility of closure by any sensor activation and are closed providing the activation is at a later time than the time of the upper portion of the window.) In this situation two separate vehicle tracks are continued along.

Figure 6f illustrates a rare occurrence. Two separate trajectories are so close that valid strips in later windows create overlapping windows on a subsequent sensor. When this occurs, one large window is created and given the identification of the lower vehicle track. Any previous admissible strips of the upper vehicle track are transferred to the lower vehicle track.

IV. HOW TO USE THE PATTERN DETECTION ALGORITHM

SPECIFYING THE INPUT DATA

A datum requiring a decimal point is shown by (D); a datum requiring an integer value is shown by (I). All data entries must fall within the column limitations specified. All integers must be right justified. Parenthetic symbols are the internal symbols in the program.

Card 1--Comments

Any comments the user desires to make to identify the run.

Card 2--Input Parameters

Cols.		
1-2	Integer 1 for identification.	(I)
3-10	Westbound average convoy velocity in kilometers per hour (AVGVEL(1)).	(D)
11-20	Westbound maximum convoy velocity in kilometers per hour (BBWND(1)).	(D)
21-30	Westbound minimum convoy velocity in kilometers per hour (UPWND(1)).	(D)
31-40	Eastbound average convoy velocity in kilometers per hour (AVGVEL(2)).	(D)
41-50	Eastbound maximum convoy velocity in kilometers per hour (BBWND(2)).	(D)
51-60	Eastbound minimum convoy velocity in kilometers per hour (UPWND(2)).	(D)
61-70	β, maximum time in minutes permitted between detections in a valid strip (BETA).	(D)
71-80	C, weight below which a sensor is considered for elimination from the string (CSENS).	(D)

Card 3--Input Parameters (continued)

<u>Cols</u>.

1-10	W, a lower bound on sensor weights. The sum of the sensor weights for those sensors contributing admissible strips must be greater than W to confirm a trajectory (WCAP).	(D)
11-20	Road segment length in meters (SEGLNT).	(D)
21-30	M, the percent of live sensors required to contribute admissible strips for trajectory confirmation (PCTSEN).	
31-40	B, the number of trajectories to be confirmed (maximum of 50) before updating weights (NB).	(I)
41-50	D, number of consecutive time periods that a sensor weight is below C before it is eliminated from the string (ND).	(I)
51-60	ω, the minimum number of detections, no more than β minutes apart, required to define a valid strip (IWCNT).	(I)
61-70	The number of sensors (maximum of 15) in the string (NSENSR).	(I)
71-78	When requesting SC-4060 graphs, specify the number of simulation minutes (multiples of ten minutes only) to be portrayed on each graph (GRAPH).	(D)
79-80	If graphs are desired, enter 1; otherwise, 0 (KAGRAF).	(I)

Card 4--Input Parameters (continued)

<u>Cols</u>.

1-2	If graphs are desired and you wish to see truck detections, false alarms, and windows separately identified, enter 1. If the graph is to show only all impulses with no windows, enter 0 (NOWIND).	(I)
3-8	ρ, the back weight for the smoothing function (RHO).	(D)

Card 5--Distance between Sensors

Cols.		
1-2	The integer 2 to identify the card.	(I)
3-10	Distance between the 1st sensor and the 2d in meters.	(D)
11-20	Between 2d and 3d.	(D)
21-30	Between 3d and 4th.	(D)
31-40	Between 4th and 5th.	(D)
41-50	Between 5th and 6th.	(D)
51-60	Between 6th and 7th.	(D)
61-70	Between 7th and 8th.	(D)
71-80	Between 8th and 9th.	(D)

If there are more than 9, start over on a new card (Cols. 1-10, 11-20, 21-30, etc.). Caution: The sum of the distances must equal precisely the length of the road segment.

A SAMPLE INPUT DECK

The input deck (shown in Appendix E) is specified in such a way that the detection algorithm will use the output information of the simulation that was stored on tape from the example shown in Appendix B. The following figure is an excerpt from the deck in Appendix E:

Col.2 Col.80

```
 THIS IS AN EXAMPLE OF THE PATTERN DETECT. ALGOR. USING SIMULATED DATA  2/12/72
1     24.0     35.0   15.0        24.0         35.0        15.0        0.40       0.05
        0.5  1000.0      .67                5           3           3           5   60.0  1
0 1.0
2     250.0      250.0      250.0      250.0
```

If the pattern detection is to be run with data other than provided by the simulation, the input tape of activations must be formatted as shown in Appendix D1 for actual data or in Appendix D2 for experimental data.

INTERPRETING THE PROGRAM OUTPUT

Figure 7 shows the first page of the output, which is a summary of all the input data. Notice on the fifth line that at least 0.67 of the sensors in the string are needed to confirm a vehicle track. The user should be aware that the computation taking place involves truncation of the fractional part of the number. Thus $0.67 \times 5 = 3$. This is important, for if 0.66 were specified and two sensors were dropped from the string as unreliable, then $0.66 \times 3 = 1$, and the program would halt with an error message, since at least two sensors are obviously required to identify a track direction. The eighth line refers to dropping hypoactive sensors, meaning any unreliable sensors, as determined by the adaptive logic. GRAPHING CONTROL: 1 indicates that there will be graphs produced. A zero would indicate no graphs. WINDOWS ON (=1) indicates that the graphs will discriminate between vehicle detections and false alarms, will display all windows, and will display confirmed vehicle tracks. The smoothing constant is the back weight ρ. The distance between sensors is shown and will always have 0.0 for the first sensor.

```
**********************************************************************************************************

                   *** PATTERN RECOGNITICN FOR VEHICLE FLOW PAST SENSORS ***

          THIS IS AN EXAMPLE OF THE PATTERN DETECT. ALGOR. USING SIMULATED DATA  2/12/72

WESTBOUND : AVG. VELOCITY =     24.00 KM/HR.     MAXIMUM VELOCITY =  35.00 K/HR    MINIMUM VELOCITY =  15.00

EASTBOUND: AVG. VELOCITY =     24.00 KM/HR.     MAXIMUM VELOCITY =  35.00 K/HR    MINIMUM VELOCITY  =  15.00 K/HR

THERE ARE   5 SENSORS ON A ROAD SEGMENT OF  1000.00 M.  AT LEAST .670 ARE NEEDED TO CONFIRM TRAJECTORIES

A VALID STRIP CONTAINS AT LEAST   3 DETECTIONS NO MORE THAN  0.400 MIN. APART

  5 TRAJ MUST BE CONFIRMED BEFORE UPDATING WEIGHTS. ANY SENSOR HAVING A WT. BELOW   0.050
IS HYPO-ACTIVE AND WILL BE DROPPED FROM THE STRING AFTER   3 CONSECUTIVE PERIODS.

THE SUM OF WEIGHTS MUST BE GREATER THAN   0.500 TO ACCEPT A TRAJECTORY CONFIRMATION.

GRAPHING CONTROL :  1   MINUTES PER GRAPH =    60.00   WINDOWS ON (=1) =    0

THE SMOOTHING CONSTANT IS = 1.0000

        SENSOR       DISTANCE(M.)


           1               0.0

           2             250.00

           3             250.00

           4             250.00

           5             250.00
```

Fig.7 — Summary of input data

Figures 8a and 8b show an example of output that gives all relevant information on the progress of the algorithm. If input data are from the simulation, information is shown on convoys. The SENSR column indicates the sensor being described. The CLOSE-OPEN columns show the time of the upper and lower portions of the window on the sensor. For example, the first entry shows a window was opened at 0.764 minutes and closed at 1.335 minutes for sensor 4.

The column labeled DIR shows the direction of the vehicle track. 1 is eastbound and 2 is westbound. TRAJ NO. shows the internal trajectory number assigned to the vehicle track. Since vehicle tracks can be in two directions, there is always a westbound and eastbound vehicle track with the same number. An interested user could follow the progress of a vehicle track using its number and directions. For example, trajectory 1, direction 1, opened windows on sensor 2 at 0.986 minutes, on sensor 3 at 1.525 minutes, on sensor 4 at 2.086 minutes, and on sensor 5 at 2.711 minutes. A vehicle track was confirmed at 3.283 minutes for this trajectory, which happened to be a convoy of five trucks (NRCNDT(5)).

The convoy information portion of the printout (for simulation data only) shows the time a convoy enters and leaves the string, and the number of trucks in the convoy. The CONVOY NO. and DIR columns identify the particular convoy, and the TRUCKS column shows the number of trucks in the convoy. If there is no entry in this column, it means that a convoy is leaving the string. For example, the first truck of convoy 1, direction 1, entered the string at time 0.0 with five trucks, and the last truck of the convoy left the string at 3.183 minutes.

The algorithm is able to tell if a particular convoy is detected by seeing if there is any overlap between the last trajectory window and the convoy transit time. For the example we have been following, the convoy left the string at 3.183 minutes, and the last window of the trajectory opened at 2.711 minutes and closed at 3.283 minutes. Thus the algorithm assumes that the confirmed vehicle track is for that convoy.

Figure 8b shows that, as requested, weights are updated after every five trajectory confirmations. The number of valid strips and

```
        WINDOW CLOSED TIMES
SENSOR    CLOSE        OPEN      DIR   TRAJ NO.



  4       1.335       0.704       2       1
  2       1.950       0.906       1       1
  3       1.960       1.309       2       1


  3       2.097       1.525       1       1
  4       2.656       2.086       1       1
  2       2.680       2.086       2       1
  5       3.283       2.711       1       1

TRAJ. CONFIRMED AT     3.283 DIR.  1 SENSOR    5
NRCNDT(5) = 1
  3       3.511       2.939       1       2
  4       3.611       3.040       2       2
  1       3.511       2.939       2       1

TRAJ. CONFIRMED AT     3.511 DIR.  2 SENSOR    1
NRCNDT(5) = 2
  2       4.113       3.542       1       3
  3       4.738       4.167       1       3
  4       4.136       3.564       1       2
  5       4.761       4.189       1       2
  3       4.236       3.665       2       2
  2       4.861       4.290       2       2
  4       5.363       4.792       1       3
  5       5.988       5.417       1       3
  1       5.486       4.915       2       2
  2      10.930      10.359       1       4
  3      11.092      10.521       1       4
  4      11.717      11.146       1       4
  5      12.342      11.771       1       4

  4      15.555      14.983       2       3
  4      16.751      16.179       1       5
  3      16.180      15.608       2       3
  2      16.751      16.179       2       3

  1      17.451      16.879       2       3

TRAJ. CONFIRMED AT    17.652 DIR.  2 SENSOR    1
NRCNDT(6) = 1
  2      18.240      17.668       1       7
  3      17.451      16.879       1       6
  4      18.076      17.504       1       6
  5      17.376      16.804       1       5
  3      18.865      18.293       1       7
  5      18.701      18.129       1       6
  4      19.490      18.918       1       7
  5      20.115      19.543       1       7

  4      22.768      22.197       2       4

  3      23.393      22.822       2       4
  4      23.718      23.147       1       9
  2      23.967      23.396       1       8
  5      24.343      23.772       1       9
  2      23.718      23.147       2       4
```

```
CONVOY ENTERS OR LEAVES AND TRACKS STRING
  CONVOY NO.   DIR       TIME    TRACKS
      1         1        0.0       5
      1         2        0.0       5
      2         1       22.427     5
      2         2       13.956     6


      1         1        3.180
      1         2        3.566

      3         2       21.506     2

      2         2       17.941

      4         2       43.623     4

      3         2       24.456
```

Fig.8a — Output on progress of algorithm

```
 3       24.496      23.924     1      8
 2       25.107      24.535     1     10
 4       25.121      24.549     1      8
 1       24.496      23.924     2      4

TRAJ. CONFIRMED AT    24.496 DIR.  2 SENSOR    1
NRCNDT(2) = 1
 5       25.746      25.174     1      8
 3       25.732      25.168     1     10
 4       26.357      25.785     1     10
 5       26.982      26.410     1     10
 3       27.483      26.912     1     11
 4       28.108      27.537     1     11
 5       28.733      28.162     1     11

 2       33.008      33.237     1     12

 3       34.432      33.861     1     12
 4       35.026      34.455     1     12
 2       35.026      34.455     2      5
 2       35.100      34.609     1     13
 3       35.805      35.234     1     13
 4       36.428      35.859     1     13
 5       35.651      35.080     1     12

TRAJ. CONFIRMED AT    35.651 DIR.  1 SENSOR    5
NRCNDT(3) = 1
VALIDS      8   7   4   0   5
ASTI'S      5   5   5   0   5
THE UPDATED W(I) :   0.250 0.250 0.250 0.0   0.250
THE     WPRIME(I) :   0.200 0.200 0.200 0.200 0.200
 4       36.019      35.448     2      6
 1       35.651      35.080     2      5
 3       36.644      36.073     2      6
 5       37.055      36.484     1     13
 2       37.269      36.698     2      6
 1       37.894      37.323     2      6
 4       39.740      39.169     2      7
 3       40.365      39.794     2      7
 4       41.044      40.473     1     14
 2       41.044      40.473     2      7
 5       41.669      41.098     1     14
 1       41.669      41.098     2      7

 4       44.098      43.527     1     15
 2       44.098      43.527     2      8
 5       44.723      44.152     1     15
 4       44.994      44.422     2      9
 1       44.723      44.152     2      8


 3       45.619      45.047     2      9
 3       46.624      46.052     1     17
 4       45.976      45.404     1     16
 5       46.401      46.029     1     16
 2       45.976      45.404     2      9
 1       46.624      46.052     2      9

TRAJ. CONFIRMED AT    46.837 DIR.  2 SENSOR    1
NRCNDT(4) = 1
 2       47.154      46.583     1     18
 4       47.249      46.677     1     17


 3       47.779      47.208     1     18
 4       48.404      47.833     1     18
```

```
 3     1     48.961     2
 2     1     35.344

 5     2     64.129     2

 4     2     46.644
 4     1     47.806     2

 5     1     93.699     6
 3     1     48.846
```

Fig.8b — Output on progress of algorithm

ASTIs on each sensor are shown, as are the updated w_i and w_i' (resulting from smoothing). In this particular example no sensors were dropped from the string; if any had been dropped, they would be shown at this point.

Figure 9 is an example of summary output provided on convoy detections. This has meaning only when input data come from the simulation model or experimental data in the format shown in Appendix D2. For each convoy size the number of convoys generated and detected are shown. We see that of the 20 generated convoys 17 were detected by the algorithm. There were 18 confirmed trajectories. Thus one was a phantom, most likely resulting from a combination of truck detections and false alarms.

*** CONVOY DETECTION SUMMARY ***

CONVOY SIZE	NO. GENERATED	NO. DETECTED
1	2	1
2	6	4
3	2	2
4	2	2
5	5	5
6	3	3
7	0	0
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0
15	0	0
16	0	0
17	0	0
18	0	0
19	0	0
20	0	0
TOTALS :	20	17

PHANTOM TRAJ = 1 TOTAL CONFIRMED = 18

Fig. 9 — Summary of convoy detections

A summary of unsmoothed sensor weights at each time period is shown in Fig. 10. The mean and standard deviations of the weights are shown for each sensor. A similar summary will follow this one for smoothed weights containing similar information.

Figure 11 is an example of the optional graphic output of activations on each sensor with no discrimination as to type of activation (false alarm or vehicle), and without windows. We have found this output useful in that it is similar to what a human observer would see.

*** SENSOR WEIGHTS BY TIME PERIOD ***

TIME PERIOD	SENSOR 1	2	3	4	5
1	0.200	0.200	0.200	0.2C0	0.200
2	0.250	0.250	0.250	0.0	0.250
3	0.278	0.278	0.167	0.0	0.278
MEAN :	0.2426	0.2426	0.2056	0.0667	0.2426
STD DEV :	0.0394	0.0394	0.0419	0.1155	0.0394

(THESE WEIGHTS ARE NOT SMOOTHED)

Fig. 10 — Summary of sensor weights

A comparison may thus be made between the accuracy of humans and that of the algorithm in detecting vehicle tracks.

Figure 12 is an example of the same graphic output as is shown in Fig. 11 but with discrimination of activations, windows, convoy midpoint passage, and trajectory confirmation shown. The + sign represents activation caused by vehicles, and the • symbol those activations caused by false alarms. The + signs to the right or left of the sensor represent passage of the first and last truck of the convoy past the sensor midpoint. The $\sqcap \atop \sqcup$ symbols represent windows for conjectured vehicle tracks moving to the left, and the $\wedge \atop \vee$ symbols, windows of conjectured vehicle tracks moving to the right. The → symbol represents vehicle track confirmation.

INTERPRETING ERROR MESSAGES

Correct the input card for the following errors:

Error	
5	ID on card 2 is not 1 in Col. 2.
10	Average velocity of westbound convoys is less than or equal 0 (cc* 3-10, card 2).
15	Average velocity of eastbound convoys is less than or equal 0 (cc 31-40, card 2).
20	Maximum velocity of westbound convoys is less than or equal 0 (cc 11-20, card 2).
25	Maximum velocity of eastbound convoys is less than or equal 0 (cc 41-50, card 2).

*cc means card column.

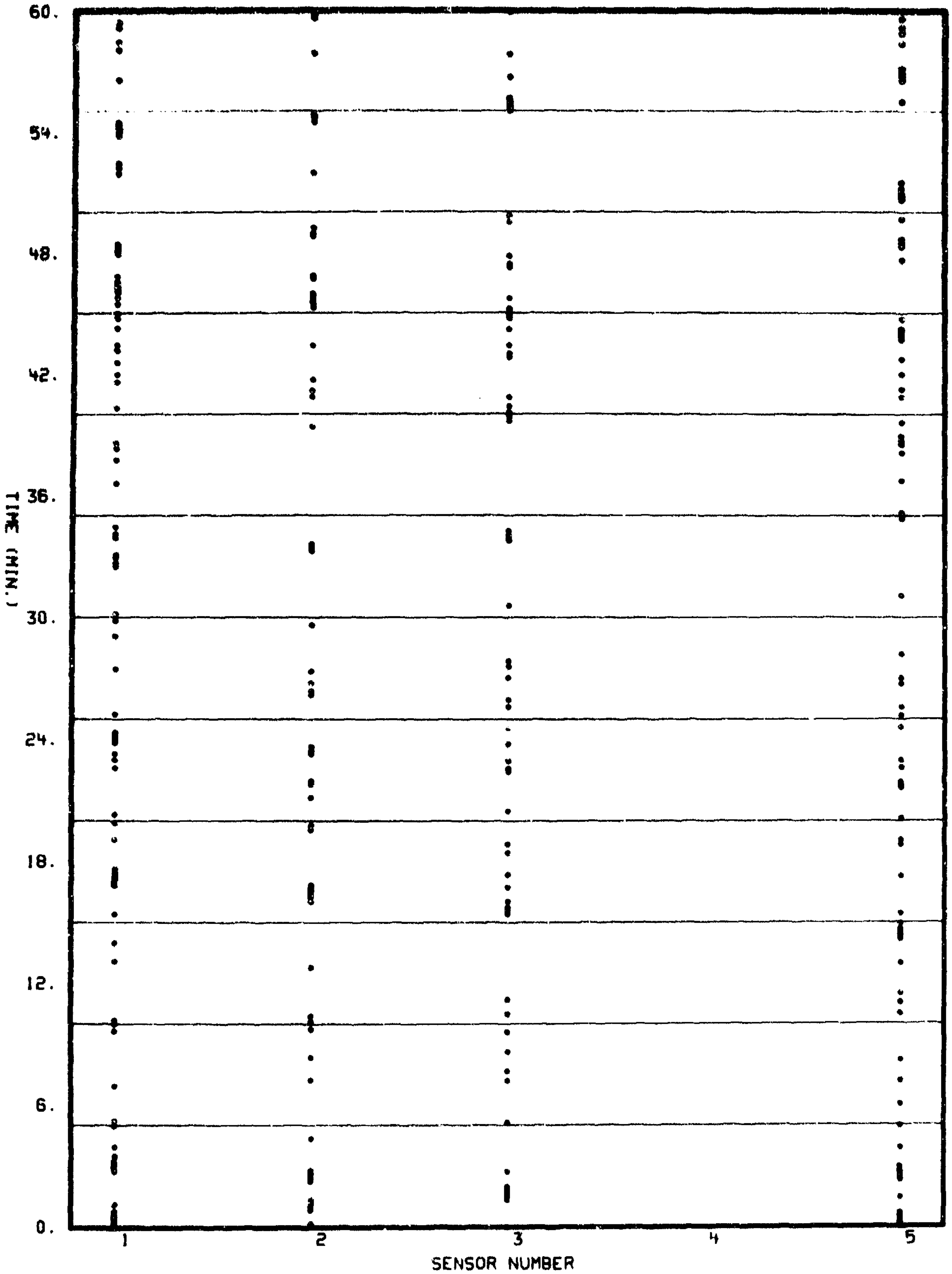


Fig. 11 — S-C 4060 graphic output (no windows shown)

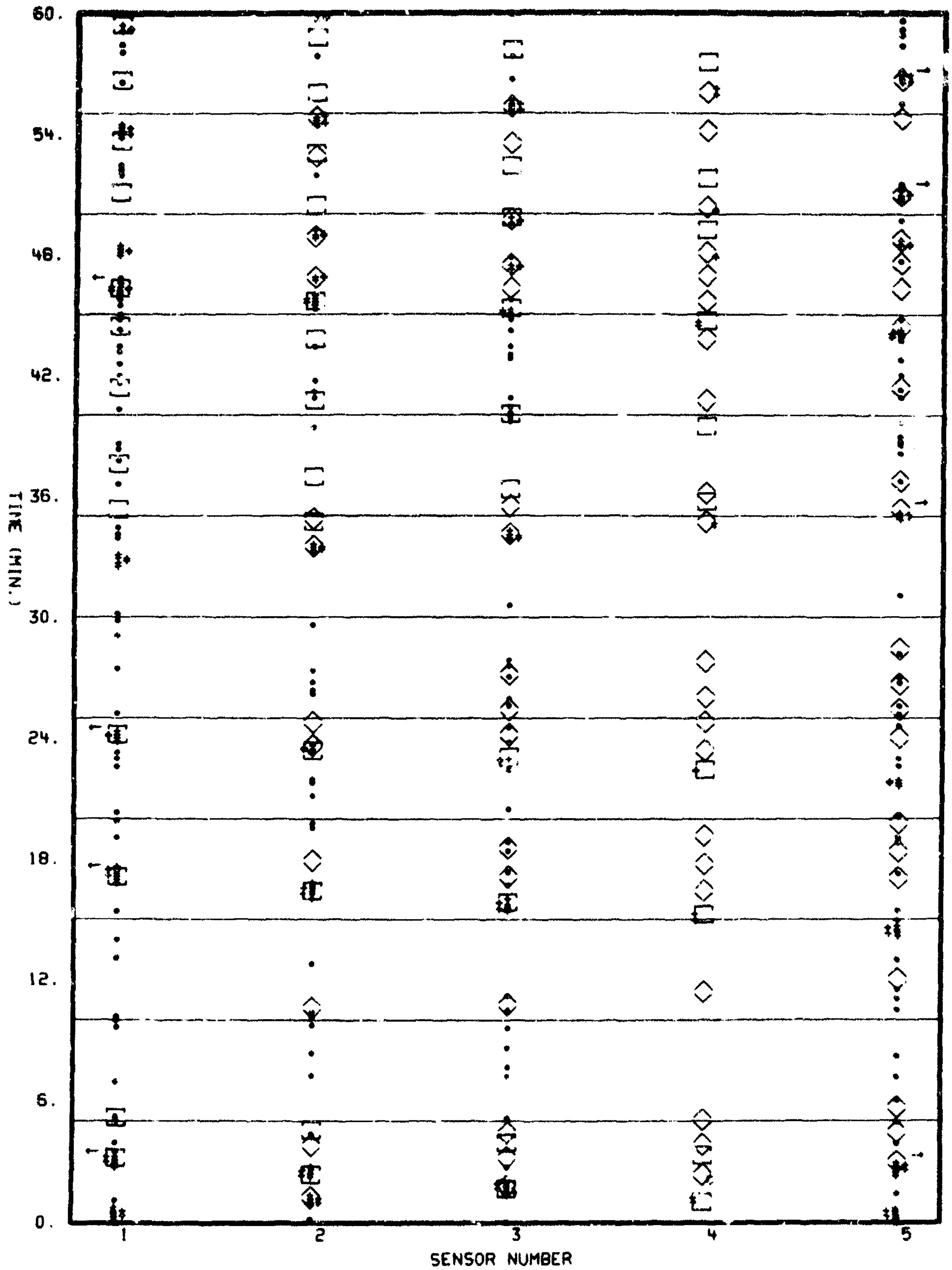


Fig. 12 — S-C 4060 graphic output (windows shown)

Error	
30	Minimum velocity of westbound convoys is less than or equal 0 (cc 21-30, card 2).
35	Minimum velocity of eastbound convoys is less than or equal 0.
40	β is less than or equal 0 (cc 61-70, card 2).
45	C is less than or equal 0 (cc 71-80, card 2).
50	W is less than or equal 0 (cc 1-10, card 3).
55	Road segment is less than or equal 0 (cc 11-20, card 3).
60	M results in less than 2 sensors (cc 21-30, card 3).
65	B is less than or equal 0 (cc 31-40, card 3).
70	D is less than or equal 0 (cc 51-60, card 3).
72	The number of sensors in the string is greater than the maximum permissible (cc 61-70, card 3).
75	ω is less than or equal 0 (cc 51-60, card 3).
80	The sum of the distances between sensors does not exactly sum to the road segment length (card 5).
117	The user has asked for more minutes of simulated time than a graph can display. Reduce the number in cc 71-78, card 3. (Must be a multiple of 10.)

The following errors are catastrophic, and processing halts unless otherwise stated. Many call for diagnosis by the maintenance programmer. We have never experienced these types, but they were included because if any were to be encountered, the results would be catastrophic.

Error	
200	A value in the IDROP array has gone negative. The value, the sensor number, and simulated time the error was detected are shown. The maintenance programmer must diagnose.
210	Same as 230 but in the CHKWIN routine.
220	Same as 260 but in the CHKWIN routine.
230	A value in the IDROP array has gone negative; it is detected in the CHKOVL routine. The sensor number and value are shown. The maintenance programmer must diagnose.

<u>Error</u>

Error	
240	The direction constant K is improper; it is detected in the CHKOVL routine. The value is shown. The maintenance programmer must diagnose.
260	A value of the NOPENT array has been detected as negative in the CHKOVL routine. The trajectory number, direction number, and value are shown. The maintenance programmer must diagnose.
270	A value of the NASTC array has been detected as negative in the CHKOVL routine. The values of NTJ, K, I2, and NASTC are shown. The maintenance programmer must diagnose.
300	The number of trucks in a convoy has been read in as a negative value from the external device (tape or disk). Thus the input data are faulty. The negative value convoy number and time of activation are shown. This error occurred in the READ routine.
410	A value in . IDROP array has gone negative, as detected in the TRAJCM routine. The value, the sensor number, and KL are shown. The maintenance programmer must diagnose.
500	Same as error 260 but in the CHKADM routine.
510	Same as error 410 but in the CHKADM routine.
520	The number of cells available in the window array plus the next available storage cell less one is greater than the cell number of the first open window. The values of the first open window cell number, the next available storage cell, and I and K are shown. The error occurred in the CHKOVL routine. The maintenance programmer must diagnose.
530	A value in the NAVLID array has gone negative in the CHKOVL routine. The value and I, J, and K are shown. The maintenance programmer must diagnose.

The following message can also occur:

THERE ARE LESS THAN 'x' SENSORS IN THE STRING. THERE ARE 'y'.

This error message is followed by a list of all sensors dropped from the string (signified by 1). The number of live sensors has gone below that asked for in cc 21-30, card 3.

ADDITIONAL INFORMATION

Appendix D1 shows the format of input data to the algorithm that come from sources other than the simulation model. Appendix E shows a fully setup data deck for executing the algorithm on the Rand IBM 360/65 installation from a data tape resulting from the simulation model of Appendix B. Appendix F contains the full source listing of the pattern detection algorithm.

Appendix A

RANDOM NUMBER SEEDS FOR THE SIMULATION MODEL

The following is a list of random number seeds for use in executing the simulation model:

(1) 2116429
(2) 8108509
(3) 4774245
(4) 1797929
(5) 4810853
(6) 6837431
(7) 9643937
(8) 1517245
(9) 1217421
(10) 6184335

Appendix B

INPUT DATA DECK FOR THE SIMULATION MODEL

The following is a fully setup deck to run the simulation model on the Rand IBM 360/65 computer installation. The output data will be saved on tape number 002325 for later use by the pattern detection algorithm.

```
//C4300*03  JOB  (5772,1000,120),'ANTHONY P. CIERVO',CLASS=A
//GO   EXEC PGM=CNVSIM,REGION=110K
//STEPLIB  DD  DSN=B4562.LIB3,DISP=SHR
//GO.SIMU02  DD  SYSOUT=B,DCB=(RECFM=FB,LRECL=80,BLKSIZE=800,BUFNO=1)
//GO.SIMU03 DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330,BUFNO=1)
//GO.SIMU05  DD  DDNAME=SYSIN
//GO.SIMU08  DD UNIT=TAPE,DSN=B4562,VOL=SER=002325,
//  DCB=(RECFM=VB,BLKSIZE=2404,LRECL=24),
//  DISP=(NEW,KEEP)
//GO.SIMU17  DD  DISP=SHR,DSN=SYS1.SIM2PERR
//GO.SYSIN  DD  *
     THIS IS AN EXAMPLE OF THE SIMULATION MODEL  2 DEC 1972
   1  20  1000.0  1000.0  5  50  250.0  0.2  20.0  0.15  40.0  1000.0 B108509
  0.0  1.0  0
      0.1  1   0.3  2   0.4  3   0.6  4   0.9  5   1.0  6  *
           0.1  24.0  20.0   35.0   15.0  WEST
           0.06 24.0  20.0   35.0   15.5  EAST
  0.5  1.5   6.0  0.5  0.2   3.0  0.5  0.01  0.05
/*
//
```

Appendix C

THE SOURCE LISTING OF THE SIMULATION MODEL

```
OLD
 PREAMBLE                                     '', M.HERMAN  8/12/71                PREB  10
  LAST COLUMN IS 72                                        ''                      PREB  12
NORMALLY MODE IS INTEGER
  THE SYSTEM HAS A TRUCKS RANDOM STEP VARIABLE IN ARRAY 1 '' SIZE CNVYS PREB  14
  THE SYSTEM HAS A FT.RUCKS IN ARRAY 1  '' PHONEY TO PRINT TRUCKS       PREB  16
NORMALLY MODE IS  REAL
   DEFINE GAMMAJ.F, BETAJ.F AS REAL FUNCTIONS                           PREB  30
 PERMANENT ENTITIES                                                     PREB  40
   EVERY  ALRM.LEVEL HAS A LAMDA '' AVG. TIME BTWEEN FALSE ALARMS (MIN) PREB  50
                     AND A UB    '' UPPER BOUND ON UNIFORM DIST OF      PREB  60
                                 '' BURST LENGTH (MIN)                  PREB  70
                     AND A LB    '' LOWER BOUND OF BURST (MIN)          PREB  80
                     AND AN OFF.PNT '' POINT TO THE ALARM OFF EVENT OF  PREB  90
                                 '' THE ALARM LEVEL                     PREB 100
                     AND A RATE     '' AVG. TIME BETWEEN OCCURANCES OF  PREB 110
                                 '' THIS LEVEL                          PREB 120
                      AND AN ON.ALARM.TIME '' TIME TURNED ON            PREB 122
                      AND AN ON.COUNTER    '' NBR. OF TIMES ON          PREB 123
                      AND A TOTAL.ON.TIME  '' TOTAL TIME LEVEL IS ON    PREB 124
                      AND A TIME.BETWEEN   '' TIME BTWN LEADING EDGES   PREB 125
   DEFINE OFF.PNT AS AN INTEGER VARIABLE                                PREB 130
     DEFINE ON.COUNTER AS AN INTEGER VARIABLE                           PREB. 132
   EVERY SENSOR HAS AN ON.TIME     '' TIME ITS READY FOR A TRANSMISSION PREB 140
                AND A DELTAS       '' DISTANCE BETWEEN INTEGRATION INCR. PREB 150
                AND A BASE         '' DISTANCE TO TRAVERSE SENSOR       PREB 160
                AND A COEFF        '' YIELDS INCR. AREA WHEN MULTIPLIED PREB 170
                                   '' BY TRUCK POSITION                 PREB 180
                AND A KILL.SIG     '' SIGNALS IF SENSOR WAS DESTROYED ON PREB 190
                                   '' IMPACT  1 . DEAD  0 - LIVE        PREB 200
                AND A PRINT.POSITION  '' FOR THE PRINT MAP OF DETECTIONSPREB 210
                 AND A LEVEL1.COUNT   ''COUNTS LEVEL 1 ALARMS           PREB 211
                 AND A LEVEL2.COUNT   ''COUNTS LEVEL 2 ALARMS           PREB 212
                 AND A LEVEL3.COUNT   ''COUNTS LEVEL 3 ALARMS           PREB 213
                 AND A DETECT.TRUCK   ''COUNTS TRUCK DETECTIONS         PREB 217
              DEFINE PRINT.POSITION AND KILL.SIG AS INTEGER VARIABLES   PREB 220
               DEFINE LEVEL1.COUNT  , LEVEL2.COUNT  , LEVEL3.COUNT   AS PREB 224
                    INTEGER VARIABLES                                   PREB 225
    EVERY ROUTE.DIRECTION HAS AN UPPER.BND '' ON VELOCITY BTWN SENSORS  PREB 230
                         AND  A LOWER.BND '' ON VELOCITY BTWN SENSORS   PREB 240
                         AND  A K1  '' PARAMETER OF A 0 - 1 BETA DIST   PREB 250
                         AND  A K2  '' PARAMETER OF A 0 - 1 BETA DIST   PREB 260
                         AND  A DIR.SYMBOL  '' ALPHA IDENT OF DIRECTIONPREB 270
                         AND  A CNV.RATE    '' MEAN TIME BETWEEN CNVYS PREB 280
              DEFINE DIR.SYMBOL AS AN ALPHA VARIABLE                    PREB 290
 TEMPORARY ENTITIES                                                     PREB 300
    EVERY FAKE HAS A S.TRUCKS IN WORD 3 '' ALLOWS ACCESS TO TRUCKS ARRAYPREB 304
              DEFINE S.TRUCKS AS AN INTEGER VARIABLE                    PREB 305
    EVERY CONVOY HAS A NBR.OF.TRUCKS                                    PREB 310
                 AND A SPACING        ''DIST. BETWEEN TRUCKS (M.)       PREB 320
                 AND A VELOCITY       '' CURRENT SPEED(M/MIN)           PREB 330
                 AND A SENSR.NBR      '' SENSOR THE LEAD TRUCK IS IN    PREB 340
                 AND A INC.IND        '' POINTS TO NEXT.SENSOR EVENT    PREB 350
                                      '' FOR THIS CONVOY                PREB 360
                 AND A CNV.LENGTH     '' CURRENT LENGTH OF THE CONVOY   PREB 370
                 AND A DIRECTION      '' IN WHICH CONVOY IS HEADED      PREB 380
```

```
                                     '' 1 - WEST , 2 - EAST          PREB 390
           AND A NBR.CONV            ''  CONVOY NUMBER (COUNT)       PREB 392
       DEFINE NBR.OF.TRUCKS, SENSR.NBR, INC.IND, NBR.CONV,           PREB 400
              DIRECTION        AS INTEGER VARIABLES                  PREB 410
EVENT NOTICES INCLUDE FALSE.ALARM,  LARM.OFF, PRNT.MAP               PREB 420
   EVERY ALARM.RATE.ON        ''CHANGES THE FALSE ALARM LEVEL        PREB 430
               HAS AN AL.TYPE  '' TYPE OF ALARM LEVEL                PREB 440
                               '' 1 - AMBIENT, 2 - MEDIUM            PREB 450
                               '' 3 - HIGH                           PREB 460
       DEFINE AL.TYPE AS AN INTEGER VARIABLE                         PREB 470
   EVERY MID.POINT.PASS       '' MARKS PASSAGE OF 1ST AND LAST TRUCK PREB 480
                              '' THROUGH THE CENTER OF SENSOR        PREB 490
               HAS A CNV.NUMBER  ''CONVOY PASSING THE MIDPOINT       PREB 500
               AND A SN.NO       '' THE SENSOR BEING EFFECTED        PREB 510
               AND A MARK        '' 1.- 1ST TRUCK  2.- LAST          PREB 514
       DEFINE CNV.NUMBER AND SN.NO AS INTEGER VARIABLES              PREB 520
   EVERY INCREMENT.CHECK          '' TESTS FOR DETECTION OF CONVOY   PREB 530
               HAS A SENS.NBR    '' SENSOR OF TEST                   PREB 540
               AND A CNVY.NBR    '' CONVOY BEING TESTED              PREB 550
               AND A DIST.TRAV   '' CURRENT DISTANCE INTO SENSORPREB 560
                                 '' OF FIRST TRUCK OF CONVOY         PREB 570
       DEFINE SENS.NBR AND CNVY.NBR AS INTEGER VARIABLES             PREB 580
   EVERY SCHED.NEXT.CONVOY   '' CREATES AND ISSUES ANOTHER CONVOY    PREB 590
               HAS A CNV.DIRECTION '' DIRECTION TO BE TRAVERSEDPREB 600
               AND A PRIER.CONVOY  '' CONVOY IN FRONT                PREB 610
       DEFINE CNV.DIRECTION AND PRIER.CONVOY AS INTEGER VARIABLES    PREB 620
   EVERY NEXT.SENSOR          '' STARTS A CONVOY THRU THE NEXT SENSOR PREB 630
               HAS A CNV.NBR     '' CONVOY BEING CONSIDERED          PREB 640
               AND A VELOC       '' SPEED FOR THE SENSOR             PREB 650
               AND A NXT.SENSR   '' THE SENSOR                       PREB 660
               AND A PRIOR.CONV  '' THE CONVOY AHEAD OF THIS ONEPREB 670
               AND A BACK.POINT  '' THE CONVOY BEHIND. 0-IF NONEPREB 675
       DEFINE CNV.NBR, NXT.SENSR ,   PRIOR.CONV  AND BACK.POINT      PREB 680
            AS INTEGER                        VARIABLES              PREB 690
   EVERY DESTROY.CONVOY         '' AFTER CONVOY LEAVES STRING DESTY IT PREB 700
               HAS A DEST.CNV  '' THE CONVOY NUMBER                  PREB 710
       DEFINE DEST.CNV AS AN INTEGER VARIABLE                        PREB 720
''                                                                   PREB 730
'' GLOBAL VARIABLES                                                  PREB 740
''                                                                   PREB 750
   DEFINE SPACE.FACTOR, ''METERS PER SPEED (M)                       PREB 760
          LAMBDA,        ''CURRENT AVG TIME BTWEEN FALSE ALARMS (MIN)  PREB 770
          DEAD.TIME,     '' TIME SENSOR IS OFF AFTER AN XMISSION (MIN) PREB 780
          MAX.TIME,      '' SIMULATION ENDS AFTER A CONVOY IS THRU   PREB 790
                         '' SENSOR FIELD AND TIME.V >  MAX.TIME (MIN)  PREB 800
          ROAD.LNGTH,    '' LENGTH OF SENSOR FIELD(M)                PREB 810
          NOMINAL.BASE, '' BASE OF TRIANGLE FOR PERFECTLY AIMED SENSOR PREB 820
          DIST.BTWN.SENSOR  '' DISTANCE BETWEEN SENSOR CENTERS       PREB 830
                 AS VARIABLES                                        PREB 840
 DEFINE I, J, K,    '' COUNTERS                                      PREB 850
       FIN.CNV  , '' NUMBER OF COMPLETED CONVOYS                     PREB 854
       PRINT.MAP, '' SIGNALS CHART PRINTING  0 - NO , 1 - YES        PREB 860
        NB.OF.CNV,''  NUMBER OF CONVOYS GENERATED                    PREB 870
       MAX.NBR.CONVS, '' SIMULATION STOPS AFTER THIS MANY CONVOYS    PREB 880
                      '' HAVE PASSED THRU THE FIELD                  PREB 890
      ERROR ,         '' ERROR COUNTER FOR INPUT DATA                PREB 900
     MAG.SENS ,       '' IND FOR MAG SENSORS 0 - NO  1 - YES         PREB 902
     DISK          '' EXTERNAL DATA SET FOR WRITING DETECTIONN       PREB 902
      ,JSAVE             '' HOLDS LARGEST CONVOY SIZE                PREB 904
    ,CONST.CNV        '' IF 1, CNVYS CREATED AT A CONSTANT RATE      PREB 906
          AS INTEGER VARIABLES                                       PREB 910
DEFINE CNV.CNTR AS AN INTEGER, 1-DIMENSIONAL ARRAY                   PREB 912
DEFINE CONVY.SIZE AS AN INTEGER, 2-DIMENSIONAL ARRAY                 PREB 914
   END                                                               PREB 920
```

```
MAIN                                          ''M. BERMAN 8/2/71
     '' READ INFORMATION ON SENSORS, SELECT SENSOR GEOMETRY
     LET DISK = 8       '' SETS EXTERNAL DATA SET NBR.
          CALL RD.SENSOR.INFO
     '' READ INFORMATION ON CONVOYS AND SCHEDULE  A CONVOY IN BOTH
     '' DIRECTIONS
     ''
          CALL RD.CONVOY.INFO
     ''
     '' READ INFORMATION ON FALSE ALARMS AND SCHEDULE ONE OF EACH TYPE
     ''
          CALL RD.FALSE.ALARM.INFO
     ''
     ''   INITIALIZE THE PRINTING OF THE SENSOR MAP
     ''
          IF PRINT.MAP IS NE 0, CALL PRINT REGARDLESS
     ''
     LET BETWEEN.V = 'TRACE'
          START SIMULATION
     ''
          END
```

```
ROUTINE BETAJ.F(K1,K2, STREAM)           '' M. BERMAN 8/5/71
 ''
 ''THIS ROUTINE CALLS GAMMAJ.F, JOHNKS METHOD.
 ''
    DEFINE K1, K2, AND X AS REAL VARIABLES
    DEFINE STREAM AS AN INTEGER VARIABLE
    IF K1<=0, LET ERR.F = 147 ELSE
    IF K2<=0, LET ERR.F = 148 ELSE
    LET X = GAMMAJ.F(1.0, K1, STREAM)
    RETURN WITH X/(X + GAMMAJ.F(1.0, K2, STREAM))
    END
```

```
SUBROUTINE TO CANCEL.FALSE.ALARM           '' M.BERMAN 8/5/71
 ''
 '' THE FALSE ALARM RATE HAS CHANGED. CANCEL THE CURRENT FALSE ALARM
 '' AND RESCHEDULE IT USING THE NEW RATE.
 ''
   CANCEL THE FALSE.ALARM
   LET TIME = EXPONENTIAL.F(LAMBDA.2) '' TIME TILL NEXT ALARM
   RESCHEDULE THE FALSE.ALARM AT TIME.V + TIME
   RETURN
   END
```

```
ROUTINE GAMMAJ.F(MEAN,K,STREAM)
''CALCULATION OF GAMMA DISTRIBUTED VARIATES BY  JOHNK'S METHOD.
''THIS ALGORITHM MUST BE USED FOR 0<K<1 INSTEAD OF GAMMA.F, AND SHOULD
''BE USED FOR NON-INTEGRAL VALUES OF K<5, ALTHOUGH IT IS 2.5 TO 3
''TIMES SLOWER THAN GAMMA.F.  FOR FURTHER DISCUSSION SEE "GENERATING
''GAMMA DISTRIBUTED VARIATES FOR COMPUTER SIMULATION MODELS",
''M. B. BERMAN,THE RAND CORPORATION, R-641-PR, FEBRUARY 1971.
DEFINE MEAN,K,KK,I,Z,A,B,D,E,X,Y, AND W AS REAL VARIABLES
DEFINE STREAM AS AN INTEGER VARIABLE
IF MEAN<=0, LET ERR.F=145 ELSE
IF K<=0,  LET ERR.F=146 ELSE
LET Z=0
LET KK=TRUNC.F(K)
LET D=K-KK
IF KK=0, GO TO BETA ELSE
LET E=1
FOR I=1 TO KK, LET E=E*RANDOM.F(STREAM)
LET Z=-LOG.E.F(E)
IF D=0,  RETURN WITH Z*(MEAN/K) ELSE
'BETA'
LET A=1/D    LET B=1/(1-D)
'NEXT'
LET X=RANDOM.F(STREAM)**A
LET Y=RANDOM.F(STREAM)**B+X
IF Y<=1, GO OUT ELSE GO TO NEXT
'OUT'
LET W=X/Y
LET Y=-LOG.E.F(RANDOM.F(STREAM))
RETURN WITH (Z+W*Y)*(MEAN/K)
END
```

```
ROUTINE TO PRINT                            '' M.BERMAN   8/12/71
  ''
  '' THIS ROUTINE PERMITS PRINTING OF A CONTINUOUS GRAPH OF FALSE
  '' ALARMS AT INTERVALS EQUAL TO THE SENSOR DEAD TIME.
  ''
     START NEW PAGE
  ''
     PRINT 1 DOUBLE LINE THUS
TIME(MIN)                                               SENSOR NUMBER

     FOR  I = 1 TO N.SENSOR, WRITE I AS (20) B (13 +  110/
                                        (1 + N.SENSOR) * I),I 2
     PRINT 1 DOUBLE LINE THUS
  0.000 I+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++I
     SCHEDULE A PRNT.MAP AT DEAD.TIME
     WRITE DEAD.TIME AS /,      D (8,3), S 1, "|", B 120, "|"
     LET LINES.V = 999999
     RETURN
     END
```

```
ROUTINE TO RD.CONVOY.INFO                         '' M. BERMAN 8/4/71
  ''
  '' THIS ROUTINE READS THE CONVOY DISTRIBUTION OF TRUCKS , THE VELOC.
  ''BETWEEN CONVOYS DIST. PARAMETERS AND SCHEDULES AN EAST AND WEST
  ''CONVOY TO INITIALIZE THE SIMULATION
  ''
     DEFINE CNV                       AS AN INTEGER VARIABLE
     CREATE A FAKE
     SKIP 2 LINES
     PRINT 1 LINE THUS
                      +++ CONVOY INFORMATION +++
     READ SPACE ''SPACING MULTIPLE'', CONST.CNV '' CONSTANT RATE FLAG
     LET SPACE.FACTOR = SPACE * 0.06
     PRINT 2 LINES WITH SPACE         THUS

 DISTANCE BETWEEN TRUCKS IN A CONVOY IS ***.**** M. FOR EACH KM/HR.
     IF SPACE.FACTOR IS LE 0.0 , ADD 1 TO ERROR
     PRINT 1 LINE THUS
     ??? ERROR IN ABOVE LINE
     ELSE
     IF CONST.CNV = 1
        PRINT 2 LINES THUS

     (NOTE: ALL CONVOYS CREATED AT CONSTANT RATE THIS RUN.)
     ELSE
     SKIP 1 CARD
     READ TRUCKS  '' READS PAIRS OF VALUES FOR STEP FUNCTION. FIRST
                  '' CUMULATIVE PROB & THEN VALUE. * AFTER LAST PAIR
     PRINT 2 LINES THUS
           DISTRIBUTION OF TRUCKS IN A CONVOY
                  TRUCKS    CUMULATIVE PROB
     LET I = FT.RUCKS
'PRT' PRINT 1 LINE WITH IVALUE.A(I) AND PROB.A(I) THUS
                  ***          *.***
     LET JSAVE = IVALUE.A(I)        '' SAVE SIZE OF LARGEST CONVOY
     LET I = S.TRUCKS(I)
     IF I = 0, GO OUT ELSE GO PRT '' PRINT VALUES
  ''
  '' READ THE CONVOY RATE AND VELOCIY DISTRIBUTION FOR EACH DIRECTION.
  '' COMPUTE K1 & K2 FOR THE BETA VELOCITY DIST.
'OUT' SKIP 2 LINES
     RESERVE CONVY.SIZE(*,*) AS 2 BY JSAVE  '' DIMENSION STATISTICS
     CREATE EVERY ROUTE.DIRECTION(2)
     FOR I = 1 TO 2    '' 1 IS WEST BOUND,  2  IS EAST BOUND
     DO
       READ RATE.CNV, MEAN     , MODE     , UPPER       , LOWER       ,
           DIR.SYMBOL(I)
     LET MEAN.VEL = MEAN * 16.6667
     LET MODE.VEL = MODE * 16.6667
     LET LOWER.BND(I) = LOWER * 16.6667
     LET UPPER.BND(I) = UPPER * 16.6667
       LET CNV.RATE(I) = 1.0/RATE.CNV  ''TIME BETWEEN CONVOYS
       IF MEAN.VEL EQ MODE.VEL '' THIS IS A SPECIAL CASE
          LET K1(I) = 2.0
          LET K2(I) = 2.0
          GO ROUND
```

```
    OTHERWISE
    LET K1(I) = (MEAN.VEL - LOWER.BND(I)) *(LOWER.BND(I) +
              UPPER.BND(I) - 2.0 * MODE.VEL)/((UPPER.BND(I) -
              LOWER.BND(I)) *(MEAN.VEL - MODE.VEL))
    LET K2(I) = K1(I) * (UPPER.BND(I) - MEAN.VEL)/(MEAN.VEL -
                                                  LOWER.BND(I))
   LET K1(I) = K1(I) + 1
   LET K2(I) = K2(I) + 1
'ROUND' PRINT 4 LINES WITH DIR.SYMBOL(I), RATE.CNV, CNV.RATE(I),
        MEAN    , MODE    , LOWER      , UPPER      , K1(I),
        K2(I) THUS
THE **** DIRECTION HAS A CONVOY RATE OF  ***.*** PER MIN. WHICH IS ONE
CONVOY EVERY *****.*** MIN.  THE AVG VELOCITY IS ****.* KM/HR.
THE MODE VELOCITY IS ****.** KM/HR.     THE SLOWEST IS ****.** KM/HR.
AND THE FASTEST IS ****.** KM/HR..  K1 IS **.*** AND K2 IS **.***
    SKIP 2 LINES
    IF RATE.CNV IS LE 0 OR MEAN.VEL IS LE 0.0 OR MODE.VEL IS LE 0.0
       OR UPPER.BND(I) IS LE 0.0 OR LOWER.BND(I) IS GE UPPER.BND(I)
       ADD 1 TO ERROR
       PRINT 1 LINE THUS
   ??? ERROR IN ABOVE 4 LINES
       GO TO NXT
    ELSE '' SCHDULE THE FIRST CONVOY
    CREATE A CONVOY CALLED CNV
          ADD 1 TO NB.OF.CNV
          ADD 1 TO CNV.CNTR(I)
          LET NBR.CONV(CNV) = CNV.CNTR(I)
    CREATE A NEXT.SENSOR CALLED INC.IND(CNV)
    LET NBR.OF.TRUCKS(CNV) = TRUCKS
   ADD 1 TO CONVY.SIZE(I, NBR.OF.TRUCKS(CNV)) '' COLLECT STATISTIC
    LET X = BETAJ.F(K1(I), K2(I), 7)  '' BETA VARIATE
    LET VELOC(INC.IND(CNV)) = X * (UPPER.BND(I) - LOWER.BND(I)) +
                                                LOWER.BND(I)
    LET CNV.NBR(INC.IND(CNV)) = CNV
    IF I = 1, LET J = 1 GO PAST  OTHERWISE LET  J= N.SENSOR
'PAST' LET NXT.SENSR(INC.IND(CNV)) = J
    LET DIRECTION(CNV) = I
    SCHEDULE THE NEXT.SENSOR CALLED INC.IND(CNV) AT 0.0
    WRITE 0.0,NBR.OF.TRUCKS(CNV), 1, 1, CNV.CNTR(I) AS BINARY
                                    USING DISK
'NXT'LOOP
   RETURN
   END
```

```
ROUTINE TO RD.FALSE.ALARM.INFO              ''  M. BERMAN  8/4/71
''
'' THIS ROUTINE READS EACH OF THE THREE FALSE ALARM LEVELS AND
'' SCHEDULE THE HIGH AND MEDIUM LEVELS FOR INITIALIZATION.
''
    LET N.ALRM.LEVEL = 3
    CREATE EVERY ALRM.LEVEL
    PRINT 2 LINES THUS
                        +++ FALSE ALARM INFORMATION +++
 ALARM LEVEL   ALARMS/MIN  MIN BURST (MIN) MAX BURST (MIN)  ARR/MIN
    READ LAM '' AMBIENT RATE
         LET LAM = LAM * N.SENSOR
    PRINT 1 LINE WITH LAM THUS
     1          ****.***
    LET LAMDA(1) = 1.0/LAM '' TIME BTWEEN ALARM AT THIS LEVEL
      LET LAMHDA = LAMDA(1)  '' START AT AMBIENT
      LET ON.ALARM.TIME(1) = 0.0
      SCHEDULE A FALSE.ALARM AT 0.0
    FOR I = 2 TO 3
    DO
      READ LAM, UB(I), LB(I), GAM  '' GAM IS THE RATE OF ARRIV FOR LVL
         LET LAM = LAM * N.SENSOR
      PRINT 1 LINE WITH I, LAM, LB(I), UB(I), GAM THUS
    **         ****.***      ***.***         ***.***       ****.***
      LET LAMDA(I) = 1.0/LAM
      LET RATE(I)  = 1.0/GAM '' TIME BETWEEN OCCURANCES OF LEVEL
      IF LAMDA(I) GE LAMDA(1) OR GAM IS LE 0.0  OR LB(I) LE 0.0 OR
         UB(I) LT LB(I) OR  LAMDA(1) LT 0.0 , ADD 1 TO ERROR
         PRINT 1 LINE THUS
    ??? ERROR IN ABOVE LINE
      ELSE '' SCHEDULE AN ALARM OF EACH TYPE
      SCHEDULE AN ALARM.RATE.ON AT .01 * I
          LET AL.TYPE(ALARM.RATE.ON) = I
    CREATE A LARM.OFF  CALLED OFF.PNT(I)
    LOOP
    RETURN
    END
```

```
ROUTINE TO RD.SENSOR.INFO                          '' M. BERMAN  8/3/71
  ''
  '' THIS ROUTINE READS THE ROAD LENGTH, NUMBER OF SENSORS, NUMBER OF
  ''INCREMENTS UNDER EACH TRIANGLE, NOMINAL BASE, AND OTHER INFORMATION
  ''PERTAINING TO SENSORS. THE ACTUAL BASE AND HEIGHT OF EACH SENSOR
  ''TRIANGLE IS SELECTED.
  ''
     START NEW PAGE
     PRINT 1 DOUBLE LINE THUS
                        +++++ SIMULATION OF TRUCK CONVOYS MOVING IN TWO DI
RECTIONS THRU A SENSOR FIELD +++++
     DEFINE COMMENT AS  A 1-DIMENSIONAL ALPHA VARIABLE '' ALLOWS THE
     RESERVE COMMENT(*) AS 20                           '' USER ONE CARD
     FOR I = 1 TO 20 , READ COMMENT(I) AS A 4           '' FOR COMMENTS
     SKIP 1 LINE
     FOR I = 1 TO 20, WRITE COMMENT(I) AS A 4
     SKIP 1 LINE
     RELEASE COMMENT(*)
     RESERVE CNV.CNTR(*) AS 2
  ''
  '' READ PRINT CONTROL, NUMBER OF CONVOYS TO BE GENERATED AND TIME TO
  '' HALT SIMULATION
     READ PRINT.MAP, MAX.NBR.CONVS, MAX.TIME '' PRINT.MAP > 0, PRINTS
     PRINT 1 DOUBLE LINE WITH MAX.TIME, MAX.NBR.CONVS THUS
  SIMULATION HALTS IF TIME EXCEEDS ****.*** MIN. OR CONVOYS GENERATED EX
CEEDS ****
     IF (MAX.NBR.CONVS IS LE 0) OR (MAX.TIME IS LE 0.0), ADD 1 TO ERROR
     PRINT 1 LINE THUS
     ??? ERROR IN ABOVE LINE
     ELSE
  ''
     READ ROAD.LNGTH, N.SENSOR , NBR.INCREMENTS, NOMINAL.BASE,
          PROB.DEAD, STD.DEV      ,DEAD.TIME, AREA, C2, SEED.V(8)
          , RAN.NBR
           IF NBR.INCREMENTS = 1   '' THE SENSORS ARE MAGNETIC
             LET MAG.SENS = 1
             LET NOMINAL.BASE = .01
             GO TO MAG
           ELSE  '' NOT MAGNETIC SENSORS
             LET MAG.SENS = 0
'MAG' SKIP 1 LINE
     PRINT 1 LINE THUS
                     +++ SENSOR PARAMETERS +++
     SKIP 1 LINE
           IF MAG.SENS NE 0  '' MAGNETIC SENSOR STRING
                  PRINT 1 LINE THUS
     THIS RUN IS FOR A MAGNETIC SENSOR STRING ONLY +++++++++++++
                  SKIP 1 LINE
           ELSE
     PRINT 2 LINES WITH ROAD.LNGTH, N.SENSOR , NOMINAL.BASE THUS
 ROUTE SEGMENT IS  *****.** M. WITH *** EQUALLY SPACED SENSORS.
 EACH WITH NOMINAL BASE OF ****.** M.
           IF MAG.SENS NE 0 '' MAGNETIC SENSOR STRING
                  PRINT 1 LINE WITH AREA THUS
     DETECTION PROBABILITY OF EACH MAGNETIC SENSOR IS  *.***
                  GO TO ONE
```

```
            ELSE                                                          RDSN 388
           IF RAN.NBR = 1.0   '' ALL EQUAL BASES                          RDSN 389
               PRINT 1 LINE WITH AREA THUS                                RDSN 390
 ALL TRIANGLES HAVE A HEIGHT OF ***.**** M.
               GO TO SIX                                                  RDSN 390
           ELSE                                                           RDSN 390
               LET C1 = AREA          '' COEFF. FOR HEIGHT                RDSN 390
               PRINT 1 LINE WITH C1. C2 THUS                              RDSN 391
 THE COEFF. FOR COMPUTING HEIGHT ARE (C1) *******.** (C2) *******.**
'SIX' PRINT 1 LINE WITH NBR.INCREMENTS THUS                               RDSN 392
 EVERY SENSOR TRIANGLE HAS **** INCREMENTS.                               RDSN 394
'ONE' IF  ROAD.LNGTH LE 0 OR N.SENSOR LE 0 OR NOMINAL.BASE LE 0 OR        RDSN 400
            AREA LE 0 OR NBR.INCREMENTS LE 0, ADD 1 TO ERROR              RDSN 410
      PRINT 1 LINE THUS                                                   RDSN 420
      ???? ERROR IN ABOVE 3 LINES                                         RDSN 430
      ELSE                                                                RDSN 440
      PRINT 3 LINES WITH PROB.DEAD, STD.DEV     , DEAD.TIME,SEED.V(8)     RDSN 460
      THUS                                                                RDSN 470
 THE PROB. THE SENSOR IS DEAD ON IMPACT IS *.***. STANDARD DEVIATION      RDSN 480
 FROM THE ROAD IS ***.*. THE DEAD TIME OF A SENSOR AFTER ACTIVATION       RDSN 490
 IS *.*** MIN. THE RANDOM SEED FOR SELECTING TRIANGLES IS *******.        RDSN 500
      IF (PROB.DEAD LT 0 OR PROB.DEAD GT 1.0) OR STD.DEV       LT 0.0     RDSN 510
                               OR DEAD.TIME LT 0.0, ADD 1 TO ERROR        RDSN 520
      PRINT 1 LINE THUS                                                   RDSN 530
      ???? ERROR IN ABOVE 3 LINES                                         RDSN 540
      ELSE                                                                RDSN 550
      IF RAN.NBR = 1.0                                                    RDSN 552
            LET HEIGHT = AREA  '' AREA IS HEIGHT FOR CONSTANT BASES       RDSN 553
            LET BASEX = NOMINAL.BASE                                      RDSN 553
         PRINT 2 LINES THUS                                               RDSN 554

      (NOTE : ALL SENSORS HAVE EQUAL BASES THIS RUN.)
      ELSE                                                                RDSN 556
  ''               SET DISTANCE BETWEEN SENSORS                           RDSN 560
      LET DIST.BTWN.SENSOR = ROAD.LNGTH/(N.SENSOR - 1)                    RDSN 570
      SKIP 2 LINES                                                        RDSN 572
      PRINT 2 LINES THUS                                                  RDSN 574
                 +++ SENSOR ATTRIBUTES +++                                RDSN 576
        SENSOR   BASE(M.)  SLOPE X DELTAS   DELTAS(M.) AREA  DEAD(=1)RDSN 578
      CREATE EACH SENSOR                                                  RDSN 580
      FOR EACH SENSOR                                                     RDSN 590
      DO                                                                  RDSN 600
           IF RANDOM.F(8) IS LE PROB.DEAD  '' SENSOR DEAD ON IMPACT       RDSN 610
              LET BASE(SENSOR) = 0.0                                      RDSN 620
              LET   KILL.SIG(SENSOR) = 1    '' SIGNALS SENSOR DEAD        RDSN 622
              GO PRNT                                                     RDSN 630
           ELSE  ''NOT KILLED ON IMPACT                                   RDSN 640
          IF MAG.SENS NE 0 '' MAGNETIC SENSOR STRING                      RDSN 642
              GO MAGS                                                     RDSN 644
          ELSE   '' CHECK FOR EQUAL BASES                                 RDSN 646
          IF RAN.NBR = 1.0       '' ALL EQUAL BASES                       RDSN 650
            GO TO TWO                                                     RDSN 660
          ELSE                   '' COMPUTE STANDARD TRIANGLES            RDSN 690
             LET DNORMAL = NORMAL.F (0.0, STD.DEV, 8)                     RDSN 690
'BACK'    IF ABS.F (DNORMAL) GT NOMINAL.BASE/2.0 '' FAR FROM ROAD         RDSN 700
             LET BASE(SENSOR) = 0.0  '' ZERO MEANS NO TRUCK DETECTION RDSN 710
            LET AREA = 0.0                                                RDSN 714
```

```
            GO PRNT                                                        RDSN 720
        ELSE  '' COMPUTE BASE                                              RDSN 730
            LET BASEX = 2.0 * (( NOMINAL.BASE/2.0) ** 2                    RDSN 740
                                - DNORMAL ** 2 ) ** .5                     RDSN 742
          IF BASEX LT 1.0 '' METER. (FOR ROUNDING PURPOSES)**              RDSN 750
            GO BACK                                                        RDSN 760
          ELSE   '' COMPUTE HEIGHT AND AREA                                RDSN 770
            LET HEIGHT = C1 /(DNORMAL ** 2 + C2)                           RDSN 780
'TWO'       LET AREA =BASEX * HEIGHT/2                                     RDSN 790
            LET BASE(SENSOR) = BASEX                                       RDSN 800
            LET DELTAS(SENSOR) = BASEX/NBR.INCREMENTS                      RDSN 810
            LET COEFF(SENSOR) = 4.0 * AREA * DELTAS(SENSOR)/               RDSN 820
                                   BASEX ** 2                              RDSN 830
            GO PRNT                                                        RDSN 840
'MAGS'         LET BASE(SENSOR) = 1.0                                      RDSN 864
               LET DELTAS(SENSOR) = .0                                     RDSN 865
               LET COEFF(SENSOR)  = AREA                                   RDSN 866
'PRNT'      LET PRINT.POSITION(SENSOR) = 13 + (110/(N.SENSOR + 1) *        RDSN 730
                                                             SENSOR)       RDSN 740
       PRINT 1 LINE WITH SENSOR, BASE(SENSOR), COEFF(SENSOR),              RDSN 750
                         DELTAS(SENSOR),AREA,KILL.SIG(SENSOR) THUS         RDSN 760
           ***     ****.**     ****.*****      **.***  ***.**    **
     LOOP                                                                  RDSN 780
     RETURN                                                                RDSN 790
     END                                                                   RDSN 800
```

```
ROUTINE FOR SUMMARY                 ''      M. BERMAN  8/12/71
   ''
   '' ALL RELAVENT STATISTICS ARE PRINTED HERE AND THE SIMULATION HALTS
   ''
      DEFINE TOT.TRUCK, TOTLTRK, GRAND.TOT AS INTEGER VARIABLES
   ''
      FOR K= 1 TO 3             '' FINISH FALSE ALARM STATISTICS
         IF LAMBA EQ LAMDA(K)
         GO OUT
         ELSE
'OUT'    LET ON.COUNTER(K) = ON.COUNTER(K) + 1
         LET TOTAL.ON.TIME(K) = TOTAL.ON.TIME(K)-+ TIME.V
                             - ON.ALARM.TIME(K)
''
'' SUMMARY OF THE SYSTEM FALSE ALARM RATES
''
      START NEW PAGE
      PRINT 2 LINES THUS
                  ++++ SYSTEM FALSE ALARMS ++++
 ALARM LEVEL  NO. OF BURSTS  AVG. BURST LENGTH(MIN)  AVG BURST ARR/MIN
      FOR I = 1 TO 3
          DO
            LET BURST = TOTAL.ON.TIME(I)/ON.COUNTER(I)
            LET TME.BTWN = TIME.BETWEEN(I)/ON.COUNTER(I)
''
            PRINT 1 LINE WITH I, ON.COUNTER(I), BURST,1.0/TME.BTWN THUS
    ***          *****              ***.****              ***.****
          LOOP
''
'' SENSOR FALSE ALARM RATES
''
      SKIP 2 LINES
       LET TOTAL = 0.0
      PRINT 2 LINES THUS
                   ++++ SENSOR FALSE ALARMS ++++
 SENSOR    ALARMS/MIN :  LEVEL 1  LEVEL 2  LEVEL 3   ALL LEVELS
      FOR EACH SENSOR
          DO
            LET TOT =(LEVEL1.COUNT(SENSOR) + LEVEL2.COUNT(SENSOR)
                       + LEVEL3.COUNT(SENSOR))/TIME.V
            PRINT 1 LINE WITH SENSOR, LEVEL1.COUNT(SENSOR)/TIME.V ,
            LEVEL2.COUNT(SENSOR)/TIME.V, LEVEL3.COUNT(SENSOR)/TIME.V,
            TOT THUS
  ***                    ****.*** ****.*** ****.***   ****.***
          LOOP
''
'' CONVOY STATISTICS
''
      START NEW PAGE
      PRINT 1 LINE THUS
              ++++ NUMBER OF CONVOYS GENERATED ++++
        LET GRAND.TOT = 0
      FOR J = 1 TO 2
          DO
            LET TOTLTRK = 0
            PRINT 2 LINES WITH DIR.SYMBOL(J) THUS
```

```
                              DIRECTION : ****
                CONVOY SIZE  FREQUENCY  TOTAL TRUCKS GENERATED
                FOR I = 1 TO JSAVE
                    DO
                      LET TOT.TRUCK = CONVY.SIZE(J,I) * I
                      LET TOTLTRK   = TOTLTRK + TOT.TRUCK
                      LET GRAND.TOT = GRAND.TOT + TOT.TRUCK
                      PRINT 1 LINE WITH I, CONVY.SIZE(J,I), TOT.TRUCK THUS
                     ***        ****            *******
                    LOOP
                PRINT 1 LINE WITH CNV.CNTR(J), TOTLTRK THUS
               TOTALS:          ****            *******
               SKIP 2 LINES
               LOOP
            PRINT 1 LINE WITH CNV.CNTR(1) + CNV.CNTR(2), GRAND.TOT THUS
          GRAND TOTALS:         ****            *******
''
'' CONVOY DETECTIONS PER SENSOR
''
            START NEW PAGE
            PRINT 2 LINES THUS
                ++++ AVERAGE CONVOY DETECTIONS BY SENSOR ++++
 SENSOR  DETECTIONS/MINUTE  DETECTIONS/CONVOY  DETECTIONS/TRUCK
   FOR EACH SENSOR
       DO
         LET DETEC.MIN = DETECT.TRUCK(SENSOR)/TIME.V
         LET DETEC.CNV = DETECT.TRUCK(SENSOR)/FIN.CNV
         LET DETEN.TRK = DETECT.TRUCK(SENSOR)/GRAND.TOT
         PRINT 1 LINE WITH SENSOR, DETEC.MIN, DETEC.CNV, DETEN.TRK THUS
  ***         ****.****          *****.****         ****.****
       LOOP
     STOP
     END
```

```
 EVENT ALARM.RATE.ON SAVING THE EVENT NOTICE        '' M. BERMAN 8/5/71
''
''  THIS EVENT SETS THE NEW ALARM LEVEL . SCHEDULES THE NEXT ONE AND
''SCHEDULES THE OFF TIME OF THE NEW LEVEL.
''
      DEFINE TYPE AS AN INTEGER VARIABLE
      LET TYPE = AL.TYPE(ALARM.RATE.ON)  '' TYPE OF THIS ALARM
      LET TIME = EXPONENTIAL.F(RATE(TYPE), 2) ''TIME OF NEXT ALARM
      SCHEDULE THE ALARM.RATE.ON AT TIME.V + TIME
      LET TIME = UNIFORM.F(LB(TYPE), UB(TYPE), 2) '' DURATION OF THIS
                                             '' LEVEL
'' IF THE CURRENT ALARM LEVEL IS AMBIENT SET THE NEW ALARM LEVEL.
'' SCHEDULE AN OFF TIME, CANCEL THE CURRENT FALSE ALARM
''
      IF LAMBDA IS EQUAL TO LAMDA(1)
            LET LAMBDA = LAMDA(TYPE)  '' SET NEW LEVEL
        LET TIME.BETWEEN(TYPE) = TIME.BETWEEN(TYPE) + TIME.V
                                 - ON.ALARM.TIME(TYPE)
        LET ON.ALARM.TIME(TYPE) = TIME.V
        LET TOTAL.ON.TIME(1) = TOTAL.ON.TIME(1) + TIME.V
                             - ON.ALARM.TIME(1)
        LET ON.COUNTER(1) = ON.COUNTER(1) + 1
            CALL CANCEL.FALSE.ALARM    '' CANCEL THE FALSE ALARM OF THE
                                       '' OLD LEVEL
'SCHD'      SCHEDULE THE  LARM.OFF CALLED OFF.PNT(TYPE) AT
                                                 TIME.V + TIME
            RETURN
      OTHERWISE '' SEE WHICH TYPE IS ON.
''
''    IF THIS TYPE IS ALREADY ON CHECK ITS OFF TIME. IF CURRENT OFF
''TIME IS LESS CANCEL IT AND RESCHED AT THE LATER TIME. IF GREATER
''IGNORE SCHEDULING A NEW OFF TIME.
''
      IF LAMBDA IS EQUAL TO LAMDA(TYPE) '' SAME TYPE IS ON
            IF  TIME.V + TIME IS LE TIME.A(OFF.PNT(TYPE))
              RETURN
            OTHERWISE '' EXTEND THE DURATION OF LAMBDA
         CANCEL THE  LARM.OFF CALLED OFF.PNT(TYPE)
          CAUSE THE  LARM.OFF CALLED OFF.PNT(TYPE) AT TIME.V + TIME
         RETURN
      OTHERWISE '' SEE IF ITS THE MEDIUM OR HIGH RATE THATS ON
 '' IF THIS IS THE MEDIUM LEVEL AND THE HIGH RATE IS ON, AND THE
 '' SCHEDULED ALARM OFF TIME IS GREATER THAN THAT ALREADY SCHEDULED
 '' FOR THE HIGH RATE, SCHED AN  ALARM OFF OTHERWISE RETURN
 ''
     IF     TYPE     IS EQUAL TO 2 '' THEN THE HIGH RATE IS ON
            IF TIME.V + TIME IS LE TIME.A(OFF.PNT(3))
            RETURN
            OTHERWISE '' SCHEDULE AN OFF TIME , EXTEND OFF TIME OR IGNOR
            IF OFF.PNT(2) IS IN THE EV.S  '' AN OFF OF THE MEDIUM TYPE
                                          '' IS ALREADY SCHEDULED
                  IF TIME.V + TIME IS LE TIME.A(OFF.PNT(2))
                     RETURN '' IT NEED NOT BE EXTENDED
                  OTHERWISE '' EXTEND IT
                  CANCEL THE  LARM.OFF CALLED OFF.PNT(2)
              RESCHEDULE THE  LARM.OFF CALLED OFF.PNT(2)
```

```
                                    AT TIME.V + TIME
           RETURN
       OTHERWISE ''.NO OFF OF THE MEDIUM TYPE IS SCHEDULED
       GO SCHD
   OTHERWISE '' THE MEDIUM RATE IS ON. CHANGE TO THE HIGH RATE. IF
             '' THE HIGH RATE WILL GO OFF AFTER THE MEDIUM RATE.
             '' CANCEL THE MEDIUM RATE ALARM OFF EVENT. ALWAYS
             '' SCHEDULE AN ALARM OFF FOR THE HIGH RATE
   LET LAMBDA = LAMDA(3)
     LET TOTAL.ON.TIME(2) = TOTAL.ON.TIME(2) + TIME.V
                          - ON.ALARM.TIME(2)
     LET ON.COUNTER(2) = ON.COUNTER(2) + 1
     LET TIME.BETWEEN(3) = TIME.BETWEEN(3) + TIME.V
                          - ON.ALARM.TIME(3)
     LET ON.ALARM.TIME(3) = TIME.V
   CALL CANCEL.FALSE.ALARM  '' AND RESCHDULE IT AT THE NEW RATE
   IF TIME.V + TIME IS LT TIME.A(OFF.PNT(2) )
      GO SCHD '' SCHEDULE AN OFF FOR THE HIGH RATE
   ELSE
   CANCEL THE  LARM.OFF CALLED OFF.PNT(2)
      GO SCHD
   END
```

```
EVENT DESTROY.CONVOY SAVING THE EVENT NOTICE ''M. BERMAN  8/12/71
 ''
 '' THIS EVENT DESTROYS THE CONVOY AND ASSOCIATED NEXT SENSOR EVENT
 '' NOTICE.  SIMULATION WILL END IF  TIME OR NUMBER OF CONVOYS HAS
 '' BEEN EXCEEDED
 ''
   DEFINE CNV, BCK.CNV     AS INTEGER VARIABLES
 ''
   LET FIN.CNV = FIN.CNV + 1  '' COUNTS COMPLRTED CONVOYS
   LET CNV = DEST.CNV(DESTROY.CONVOY)
   LET BCK.CNV = BACK.POINT(INC.IND(CNV))  '' CONVOY BEHIND THIS 1
        IF BCK.CNV NE 0
   LET PRIOR.CONV(INC.IND(BCK.CNV)) = 0 '' INDICATES NO MORE CNV
        ELSE
 ''
   WRITE TIME.V, 0, 2, DIRECTION(CNV), NBR.CONV(CNV) AS BINARY USING
                                                   DISK
   DESTROY THE NEXT.SENSOR CALLED INC.IND(CNV)
   DESTROY THE CONVOY CALLED CNV
       IF (FIN.CNV GE MAX.NBR.CONVS) OR ((TIME.V GE MAX.TIME)
                  AND (FIN.CNV EQ NB.OF.CNV))
   WRITE TIME.V + 2..1, 3, 0, 0 AS BINARY USING DISK  '' FALSE ALARM
                               '' TO CLOSE ANY REMAINING WINDOWS
   WRITE TIME.V + 4..1,3, 0, 0 AS BINARY USING DISK
   WRITE  TIME.V, 0, 9, 0, 0  AS BINARY USING DISK
        CALL SUMMARY
   ELSE
   RETURN
   END
```

```
EVENT FALSE.ALARM SAVING THE EVENT NOTICE
''
'' THIS EVENT WILL CAUSE A FALSE ALARM ON A SENSOR IF THE SENSOR IS
'' ON. IT PRINT & WRITES THE TIME OF ACTIVATION. IT RESCHEDULES THE
'' NEXT FALSE ALARM.
''
     LET I = RANDOM.F(2) * N.SENSOR  + .5  '' SENSOR NUMBER
     IF   KILL.SIG(I) IS GT 0 '' THE SENSOR WAS DESTROYED ON IMPACT
        GO SCHD
     ELSE
     IF ON.TIME(I) IS LT TIME.V '' THE SENSOR IS ON
            IF LAMBDA IS EQ LAMDA(1)
               LET OUT.F(PRINT.POSITION(I)) = "1"
               ADD 1 TO LEVEL1.COUNT(I)
               GO WRTE
            ELSE
            IF LAMBDA IS EQ LAMDA(2)
               LET OUT.F(PRINT.POSITION(I)) = "2"
               ADD 1 TO LEVEL2.COUNT(I)
               GO WRTE
            ELSE
        LET OUT.F(PRINT.POSITION(I)) = "4"
               ADD 1 TO LEVEL3.COUNT(I)
'WRTE'  WRITE TIME.V,  I,3,0,0  AS BINARY USING DISK
        LET ON.TIME(I) = TIME.V + DEAD.TIME
     REGARDLESS
'SCHD' LET TIME = EXPONENTIAL.F(LAMBDA, 3) '' TIME TILL NEXT ALARM
       SCHEDULE THE FALSE.ALARM AT TIME.V + TIME
     RETURN
     END
```

```
EVENT INCREMENT.CHECK SAVING THE EVENT NOTICE ''M. BERMAN 8/11/71
  ''
  '' THIS EVENT CHECKS FOR A CONVOY DETECTION IN A SENSORS SPHERE OF
  '' INFLUENCE. THE NEXT INCREMENT CHECK IS SCHEDULED AT DELTAT. IF
  '' THERE IS A DETECTION ITS RECORDED. NO INCREMENT CHECK WILL BE
  '' SCHEDULED IF THE LAST TRUCK OF THE CONVOY WILL BE OUT OF THE
  '' SENSORS SPHERE OF INFLUENCE.
  ''
     DEFINE CNV, SENS                 AS INTEGER VARIABLES
  ''
  ''              OBTAIN ATTRIBUTES OF THE EVENT
  ''
     LET SENS = SENS.NBR(INCREMENT.CHECK)  '' SENSOR NBR
     LET  CNV = CNVY.NBR(INCREMENT.CHECK)  '' CONVOY NBR
     LET P1ST = DIST.TRAV(INCREMENT.CHECK) '' POSITION OF 1ST TRUCK
   ''
     IF MAG.SENS NE 0  '' THE SENSOR IS MAGNETIC
          CALL MAG.DETECTIONS ( SENS, CNV, P1ST)
          RETURN
     ELSE
     LET BASE.DIST = BASE(SENS)  '' BASE LENGTH OF SENSOR
  ''
     IF ON.TIME(SENS) IS LE TIME.V  '' SENSOR CAN TRANSMIT
  ''
  '' CALCULATE THE PROBABILITY OF NOT DETECTING EACH TRUCK IN THE
  '' SENSORS SPHERE OF INFLUENCE
  ''
     LET PROB = 1.0   '' INIALIZE PROBABILITY
     LET HALF.BASE = BASE.DIST/2.0 '' 1/2  THE BASE
     LET SPAC.DIST = SPACING(CNV)  '' SPACE BETWEEN TRUCKS
     LET COEF      = COEFF (SENS)  '' YIELD INCREMENT AREA FROM DIST
     FOR I = 0 TO NBR.OF.TRUCKS(CNV) - 1
      DO
         LET TRUCK.POSITION =P1ST - I * SPAC.DIST
         IF  TRUCK.POSITION IS LE 0.0  '' IS TRUCK IN BASE
               GO OUT
         ELSE '' SEE IF ITS BEYOND BASE
         IF  TRUCK.POSITION IS GE BASE.DIST
               GO TO NEXT ''TRUCK IN CONVOY
         ELSE '' TRUCK IS WITHIN BASE
         IF TRUCK.POSITION IS GT HALF.BASE ''ON DECREASING SIDE
               LET TRUCK.POSITION = BASE.DIST - TRUCK.POSITION
         ELSE '' ITS ON THE INCREASING SIDE
         LET PROB = PROB * (1.0 - COEF * TRUCK.POSITION)
'NEXT' LOOP
'OUT'  '' SEE IF DETECTION IS MADE
   ''
    IF PROB IS LT RANDOM.F(3) '' A DETECTION IS MADE
         LET ON.TIME(SENS) = TIME.V + DEAD.TIME '' SENSOR IS NOW OFF
         LET OUT.F(PRINT.POSITION(SENS)) = DIR.SYMBOL(DIRECTION(CNV))
         WRITE TIME.V,SENS,6,DIRECTION(CNV),NBR.COPY(CNV) AS BINARY
                                 USING DISK
         ADD 1 TO DETECT.TRUCK(SENS)
         LET TIME = ON.TIME(SENS) '' TIME OF NEXT CHECK
         GO CHECK
    ELSE '' CONVOY NOT DETECTED
```

```
          LET TIME = DELTAS(SENS)/VELOCITY(CNV)  '' DELTAT NEXT CHECK INCK 500
                                        + TIME.V                      INCK 505
  ''                                                                  INCK 510
  '' SEE IF THIS IS THE LAST SENSOR IN THE STRING                     INCK 520
  ''                                                                  INCK 530
'CHECK'                                                               INCK 530
    IF (SENS = N.SENSOR AND DIRECTION(CNV) = 1) OR (SENS = 1 AND      INCK 540
                                                   DIRECTION(CNV)=2)  INCK 550
         GO TO LAST '' THE CONVOY SPEED WILL NOT CHANGE               INCK 560
    ELSE '' CHECK TO SEE IF VELOCITY WILL CHANGE BEFORE THE NEXT      INCK 570
                                          '' INCREMENT CHECK          INCK 580
     IF TIME.A(INC.IND(CNV)) IS LT TIME '' VELOCITY WILL CHANGE       INCK 590
         LET PIST = (TIME - TIME.A(INC.IND(CNV)))                     INCK 600
                   *   VELOC(INC.IND(CNV)) + (TIME.A(INC.IND(CNV))-   INCK 610
                 TIME.V) * VELOCITY(CNV) + PIST '' 1ST TRUCK          INCK 620
         LET PLAST = PIST -  (NBR.OF.TRUCKS(CNV)- 1) * SPACE.FACTOR   INCK 630
                         *   VELOC(INC.IND(CNV)) ''LAST TRUCK         INCK 640
         GO TO CHK                                                    INCK 650
    ELSE                                                              INCK 660
'LAST'LET PIST =(TIME - TIME.V) * VELOCITY(CNV) + PIST                INCK 670
    LET PLAST = PIST - CNV.LENGTH(CNV)                                INCK 680
  ''                                                                  INCK 690
  ''  NOW CHECK TO SEE IF CONVOY WILL BE IN SENSOR SPHERE OF INFLUENCE INCK 700
'CHK'                                                                 INCK 710
    IF PLAST IS GE BASE.DIST '' LAST TRUCK WILL BE OUTSIDE SENSOR     INCK 720
         LET TIME.SPL = TIME - (PLAST - BASE.DIST)/VELOCITY(CNV)      INCK 722
         WRITE TIME.SPL,SENS ,8, DIRECTION(CNV), NBR.CONV(CNV)        INCK 724
                    AS BINARY USING DISK                              INCK 726
         DESTROY THE INCREMENT.CHECK                                  INCK 730
         RETURN                                                       INCK 740
    ELSE '' IT WILL BE IN THE SENSOR AREA OF INFLUENCE                INCK 750
 ''                                                                   INCK 760
    SCHEDULE THE INCREMENT.CHECK AT TIME                              INCK 770
         LET DIST.TRAV(INCREMENT.CHECK) = PIST                        INCK 780
         RETURN                                                       INCK 790
    ELSE ''SENSOR IS OFF                                              INCK 800
    LET TIME = ON.TIME(SENS)                                          INCK 810
    GO TO CHECK                                                       INCK 820
    END                                                               INCK 830
```

```
ROUTINE FOR MAG.DETECTIONS ( SENS, CNV, NUM.TRUK ) '' M. HERMAN 12/7/71
 ''
 ''  THIS ROUTINE PROCESS MAGNETIC SENSOR DETECTIONS. THESE SENSORS
 ''  ACT AS TRIPWIRES AND DETECT WITH PROBABILITY P (COEFF(SENSOR).
 ''
   DEFINE SENS, CNV                              AS INTEGER VARIABLES
 ''
     IF (ON.TIME(SENS) LE TIME.V)   '' SENSOR CAN TRANSMIT
          AND (COEFF(SENS) GE RANDOM.F(3)) '' TRUCK WILL BE DETECTED
                 LET ON.TIME(SENS) = TIME.V + DEAD.TIME '' SENSOR OFF
                 LET OUT.F(PRINT.POSITION(SENS)) =
                                     DIR.SYMBOL(DIRECTION(CNV))
                 WRITE TIME.V, SENS, 6, DIRECTION(CNV), NBR.CONV(CNV)
                               AS BINARY USING DISK
                 ADD 1 TO DETECT.TRUCK(SENS)
    ELSE ''NO DETECTION IS MADE
           IF NUM.TRUK  EQ 1.0'' THIS IS THE LAST TRUCK IN THE CONVOY.
             LET TIMER = TIME.V + BASE(SENS)/VELOCITY(CNV)
                 WRITE TIMER , SENS, 8, DIRECTION(CNV), NBR.CONV(CNV)
                               AS BINARY USING DISK
                 DESTROY THE INCREMENT.CHECK
                 RETURN
          OTHERWISE ''SCHEDULE THE NEXT TRUCK DETECTION
                 SCHEDULE THE INCREMENT.CHECK AT TIME.V +
                                     SPACING(CNV)/VELOCITY(CNV)
                 LET DIST.TRAV(INCREMENT.CHECK) = NUM.TRUK -1.0 ''NEXT
                 RETURN                                      '' TRUCK
    END
```

```
EVENT  LARM.OFF SAVING THE EVENT NOTICE    '' M. HERMAN 8/5/71
''
'' THIS EVENT TURNS AN ALARM OF THE MEDIUM AND HIGH LEVELS OFF; IF A
'' MEDIUM OFF IS SCHEDULED THAT WILL BE THE NEW LEVEL OTHERWISE THE
'' RATE WILL DROP TO AMBIENT
''
     IF LAMBDA EQ LAMDA(2)   '' WHICH ALARM LEVEL IS ON ?
          LET I = 2          ''  LEVEL 2
          GO COLCT
     ELSE
          LET I = 3          ''  LEVEL 3
'COLCT' LET TOTAL.ON.TIME(I) =
              TOTAL.ON.TIME(I) + TIME.V - ON.ALARM.TIME(I) '' COLLECT STAT
        LET ON.COUNTER(I) = ON.COUNTER(I) + 1
     IF EV.S( I.LARM.OFF) IS EMPTY  '' NO OTHER ALARM LEVEL IS ON
         LET LAMBDA = LAMDA(1)        '' RETURN TO AMBIENT LEVEL
        LET TIME.BETWEEN(1) = TIME.BETWEEN(1)+ TIME.V - ON.ALARM.TIME(1)
        LET ON.ALARM.TIME(1) = TIME.V
         CALL CANCEL.FALSE.ALARM      '' RESCHEDULE FALSE ALARM FOR NEW
         RETURN                       '' LEVEL
     ELSE        ''SOME RATE IS ON
     IF OFF.PNT(2) IS IN THE EV.S              '' ITS THE MEDIUM RATE
         LET LAMBDA = LAMDA(2)
        LET TIME.BETWEEN(2) = TIME.BETWEEN(2)+ TIME.V - ON.ALARM.TIME(2)
        LET ON.ALARM.TIME(2) = TIME.V
         CALL CANCEL.FALSE.ALARM
         RETURN
     OTHERWISE ''ITS THE HIGH RATE
         LET LAMBDA = LAMDA(3)
        LET TIME.BETWEEN(3) = TIME.BETWEEN(3)+ TIME.V - ON.ALARM.TIME(3)
        LET ON.ALARM.TIME(3) = TIME.V
         CALL CANCEL.FALSE.ALARM
         RETURN
     END
```

```
EVENT MID.POINT.PASS SAVING THE EVENT NOTICE   '' M. BERMAN 8/12/71
  ''
  ''THIS EVENT MARKS THE TIME OF PASSAGE OF THE FIRST AND LAST TRUCK
  ''OF THE CONVOY THROUGH THE MID POINT OF THE SENSOR.
  ''
     DEFINE CNV, SENS    AS INTEGER VARIABLES
  ''
     LET CNV = CNV.NUMBER(MID.POINT.PASS)
     LET SENS= SN.NO(MID.POINT.PASS)
     LET IND3 =  MARK(MID.POINT.PASS)
  ''
     IF DIRECTION(CNV) IS EQUAL TO 1    '' CONVOY IS WEST BOUND
          LET OUT.F(PRINT.POSITION(SENS)+ 2) = "-"
          GO OUT
     ELSE  '' ITS EAST BOUND
          LET OUT.F(PRINT.POSITION(SENS) - 2) = "+"
'OUT' DESTROY THE MID.POINT.PASS
     WRITE TIME.V, SENS, 4, DIRECTION(CNV), 0 AS BINARY USING DISK
   ''
   ''CHECK TO SEE IF SENSOR IS COMPLETED (IF BASE IS 0)
   ''
       IF BASE(SENS) = 0.0 AND IND3 = 2.0
     LET TIME = TIME.V +  0.001         '' GIVES DIMENSIN TO 1 TRUCK
       WRITE TIME  , SENS, 8, DIRECTION(CNV), NBR.CONV(CNV) AS
                                           BINARY USING DISK
       REGARDLESS
     RETURN
     END
```

```
EVENT NEXT.SENSOR SAVING THE EVENT NOTICE  '' M. BERMAN 8/6/71
''
'' THIS EVENT INDICATE THAT A CONVOY HAS JUST ENTERED THE BASE OF A
'' TRIANGLE. IT WILL SELECT THE VELOCITY AT THE NEXT SENSOR AND
'' SCHEDULE THE NEXT SENSOR EVENT. IF THIS IS THE SECOND SENSOR FOR
'' EITHER DIRECTION IT WILL SCHEDULE A NEXT CONVOY. ADDITIONALLY IT
'' SCHEDULE A MID POINT PASS EVENT FOR THE FIRST AND LAST TRUCK.
''
   DEFINE CNV, SENS, NXT.SNS, DIR,SENS.POSIT, PRE.CNV,FLAG1
                          AS INTEGER VARIABLES
   LET CNV = CNV.NBR(NEXT.SENSOR)              '' CURRENT CONVOY
   LET VELCTY = VELOC(NEXT.SENSOR)             '' CURRENT VELOCITY
   LET SENS = NXT.SENSR(NEXT.SENSOR)           '' CURRENT SENSOR
   LET PRE.CNV = PRIOR.CONV(NEXT.SENSOR)       '' CONVOY IN FRONT
   LET FLAG1 = 1                       '' (0) INDICATES END OF STRING
''
''  SET ALL NEW ATTRIBUTES OF CONVOY
   LET SPACING(CNV) = SPACE.FACTOR * VELCTY  '' DIST BETWEEN TRUCKS
   LET VELOCITY(CNV) = VELCTY
   LET CNV.LENGTH(CNV) = SPACING(CNV) *(NBR.OF.TRUCKS(CNV)- 1)
   LET SENSR.NBR(CNV)   = SENS  '' FIRST TRUCK IS IN THIS SENSOR FIELD
   LET DIR              = DIRECTION(CNV) '' 1 IS WEST,2 IS EAST
   WRITE TIME.V, SENS, 7, DIR, NBR.CONV(CNV) AS BINARY USING DISK
''
''  IF THIS SENSOR IS THE LAST IN THE STRING SCHEDULE A DESTROY CONVOY
''
   IF (DIR =1 AND SENS = N.SENSOR) OR (DIR = 2 AND  SENS = 1)
        LET TIME = (BASE(SENS) + CNV.LENGTH(CNV))/ VELCTY
        SCHEDULE A DESTROY.CONVOY AT TIME.V + TIME
              LET DEST.CNV(DESTROY.CONVOY) = CNV
              LET FLAG1 = 0
        GO TO MID
   ELSE '' IF THIS IS THE SECOND SENSOR FOR A CONVOY SCHDULE THE NEXT
                     ''CONVOY. THIS PREVENT MORE THAN ONE CONVOY IN
                     ''AREA UNDER SENSOR FOR EACH TRIANGLE
   IF (SENS = 2  AND DIR = 1) OR (SENS = N.SENSOR - 1 AND DIR = 2)
        LET TIME = CNV.LENGTH(CNV)/VELCTY
       SCHEDULE A SCHED.NEXT.CONVOY AT TIME.V + TIME
              LET CNV.DIRECTION(SCHED.NEXT.CONVOY) = DIR
              LET PRIER.CONVOY(SCHED.NEXT.CONVOY) = CNV
   REGARDLESS '' OBTAIN VELOCITY AT NEXT SENSOR AND CHECK SPACING
   LET NXT.SNS = SENS + 1
   IF DIR = 2,  LET NXT.SNS = SENS - 1
   REGARDLESS
   LET X = BETAJ.F(K1(DIR), K2(DIR), 3)
   LET VEL.NXT = X * (UPPER.BND(DIR) - LOWER.BND(DIR))
                                + LOWER.BND(DIR)
'' CHECK PRIOR CONVOY TO INSURE WE WONT CATCH IT.
   IF PRE.CNV IS NE 0                             '' CONVOY AHEAD
        LET J = CNV.LENGTH(PRE.CNV)/DIST.BTWN.SENSOR + .5 ''EQUIV.
        IF DIR = 2                                     ''SENSORS
             LET J = - J
        REGARDLESS '' GET MAX SENSOR NUMBER OF LAST TRUCK IN CONVOY
        LET SENS.POSIT     = SENSR.NBR(PRE.CNV) - J
      IF (DIR=1 AND SENS.POSIT <= NXT.SNS) OR (DIR= 2 AND
                SENS.POSIT >= NXT.SNS)  '' THERE MAY BE A TRUCK UP
```

```
          THEN IF VEL.NXT IS GT VELOCITY(PRE.CNV) ''GOING TOO FAST
                   LET VEL.NXT = VELOCITY(PRE.CNV)   ''SET SPEEDS EQUAL
            OTHERWISE '' NO TRUCK IMMEDIATELY AHEAD
     ELSE '' NO CONVOY IN THE STRING GOING THE SAME DIRECTION
  '''
  '' SET THE TIME OF THE NEXT.SENSOR.EVENT
  ''
     LET TIME = (DIST.BTWN.SENSOR - BASE(NXT.SNS)/2.0 + BASE(SENS)/2.0)
                                        /VELCTY
     SCHEDULE THE NEXT.SENSOR CALLED INC.IND(CNV) AT  TIME.V + TIME
            LET NXT.SENSR(NEXT.SENSOR) = NXT.SNS
            LET VELOC(NEXT.SENSOR) = VEL.NXT
 ''
 '' SCHEDULE THE TIME THE FIRST AND LAST TRUCK OF THE CONVOY WILL PASS
 '' THE CENTER OF THE SENSOR
'MID'
     LET TIME =BASE(SENS)/(2.0 * VELCTY) '' TIME FIRST TRUCK CROSSES
     SCHEDULE  A  MID.POINT.PASS AT TIME.V-+ TIME
            LET CNV.NUMBER(MID.POINT.PASS) = CNV
            LET SN.NO(MID.POINT.PASS) = SENS
            LET MARK(MID.POINT.PASS) = 1.0
        IF FLAG1 = 0                       '' NO MORE SENSORS
            GO TIM
        ELSE
     IF DIST.BTWN.SENSOR - BASE(NXT.SNS)/2.0 GT CNV.LENGTH(CNV)''SPEED
'TIM'       LET TIME = CNV.LENGTH(CNV)/VELCTY  + TIME     ''WONT CHANGE
            GO SCHD
     ELSE '' THERE WILL BE A SPEED CHANGE
     LET TIME =(DIST.BTWN.SENSOR - BASE(NXT.SNS)/2.0)/VELCTY +
         (CNV.LENGTH(CNV) - DIST.BTWN.SENSOR + BASE(NXT.SNS)/2.0)
                               /VEL.NXT  + TIME
'SCHD'SCHEDULE A MID.POINT.PASS AT TIME.V + TIME
            LET CNV.NUMBER(MID.POINT.PASS) = CNV
            LET SN.NO(MID.POINT.PASS) = SENS
            LET MARK(MID.POINT.PASS) = 2.0
     IF BASE(SENS) IS  EQUAL TO 0.0  '' THE SENSOR CANNOT RECORD TRUCKS
            RETURN
     ELSE
     SCHEDULE AN INCREMENT.CHECK AT TIME.V + DELTAS(SENS)/(2. * VELCTY)
            LET SENS.NBR(INCREMENT.CHECK) = SENS
            LET CNVY.NBR(INCREMENT.CHECK) = CNV
          LET DIST.TRAV(INCREMENT.CHECK) = DELTAS(SENS)/2.0
                   IF MAG.SENS NE 0 '' THE SENSOR IS MAGNETIC
                     LET DIST.TRAV(INCREMENT.CHECK) = NBR.OF.TRUCKS(CNV)
                  OTHERWISE
     RETURN
     END
```

```
EVENT PRNT.MAP SAVING THE EVENT NOTICE        '' M. BERMAN   8/12/71
  ''
  ''THIS EVENT PRINTS ANY FALSE ALARM OR DETECTION THAT HAS OCCURRED
  ''IN THE INTERVAL DEAD.TIME ON ANY SENSOR
  ''
     LET TME = TIME.V + DEAD.TIME
     WRITE TME    AS /,       D (8,3), S 1,  "|", B 120, "|"
     SCHEDULE THE PRNT.MAP AT TME
     RETURN
     END
```

```
EVENT SCHED.NEXT.CONVOY SAVING THE EVENT NOTICE'' M. BERMAN 8/10/71
 ''
 '' THIS EVENT CREATES AND SCHEDULES A CONVOY TO PASS THROUGH THE
 '' SENSOR FIELD. THE VELOCITY FOR THE FIRST SENSOR IS SELECTED AND
 '' A NEXT SENSOR EVENT IS SCHEDULED FOR THE FIRST SENSOR
 ''
     DEFINE DIR,PRE.CNV, CNV, SENS,TRUK1 AS INTEGER VARIABLES
     CREATE A CONVOY CALLED CNV
     ADD 1 TO  NB.OF.CNV    ''COUNTS CONVOYS GENERATED
          IF NB.OF.CNV GT MAX.NBR.CONVS
             GO TO FIN
          ELSE
     LET DIR = CNV.DIRECTION(SCHED.NEXT.CONVOY)
     LET PRE.CNV = PRIER.CONVOY(SCHED.NEXT.CONVOY)
 ''       SELECT A NUMBER OF TRUCKS FOR THE CONVOY
     LET NBR.OF.TRUCKS(CNV) = TRUCKS
     ADD 1 TO CONVY.SIZE(DIR, NBR.OF.TRUCKS(CNV)) '' CO.CT STATISTIC
     LET DIRECTION(CNV) = DIR
     ADD 1 TO CNV.CNTR(DIR) '' ASSIGN A CONSECUTIVE NBR. TO CONVOY
     LET NBR.CONV(CNV) = CNV.CNTR(DIR)
 ''
 ''  OBTAIN THE TIME      AT THE FIRST SENSOR (NOMINAL BASE)
 ''
     LET TIME = EXPONENTIAL.F(CNV.RATE(DIR), 3)
     IF CONST.CNV = 1        '' USER ASKED FOR CONSTANT CONVY RATE
        LET TIME = CNV.RATE(DIR)
     ELSE
 ''
 ''  OBTAIN THE VELOCITY THRU THE FIRST SENSOR
 ''
     LET X =BETAJ.F(K1(DIR),K2(DIR), 3)
     LET VEL = X * (UPPER.BND(DIR) - LOWER.BND(DIR)) + LOWER.BND(DIR)
 '' CHECK TO BE SURE WE MAINTAIN AT LEAST ONE SENSOR BETWEEN CONVOYS
 ''
     IF TIME * LOWER.BND(DIR)      IS LT DIST.BTWN.SENSOR
                IF TIME.A(INC.IND(PRE.CNV)) IS LT TIME.V + TIME
                        IF VEL IS GT VELOC(INC.IND(PRE.CNV))
                               LET VEL = VELOC(INC.IND(PRE.CNV))
                               GO OUT
                        ELSE
                               GO OUT
               OTHERWISE '' CURRENT VELOCITY OF THE CONVOY WILL HOLD
                IF VEL IS GT VELOCITY(PRE.CNV)
                 LET VEL = VELOCITY(PRE.CNV)
                ELSE
     REGARDLESS
'OUT'
 ''  TIME AT THE FIRST SENSOR
 ''
     LET FIRST.BASE = BASE(1)       IF DIR = 2, LET FIRST.BASE =
                                                 BASE(N.SENSOR)
     REGARDLESS
       LET TIME = TIME + (NOMINAL.BASE - FIRST.BASE)/( 2.0 * VEL )
          IF TIME.V + TIME GT MAX.TIME
             SUBTRACT 1 FROM NB.OF.CNV
             SUBTRACT 1 FROM CONVY.SIZE(DIR,NBR.OF.TRUCKS(CNV))
```

```
            SUBTRACT 1 FROM CNV.CNTR(DIR)
            GO TO FIN
          ELSE
      SCHEDULE A NEXT.SENSOR CALLED INC.IND(CNV) AT TIME.V + TIME
      LET SENS = 1  IF DIR = 2, LET SENS = N.SENSOR  REGARDLESS
           LET NXT.SENSR(INC.IND(CNV)) = SENS
           LET CNV.NBR (INC.IND(CNV)) = CNV
           LET  VELOC  (INC.IND(CNV)) = VEL
           LET PRIOR.CONV(INC.IND(CNV)) = PRE.CNV
           LET BACK.POINT(INC.IND(PRE.CNV)) = CNV '' UPDATE THE BACK
                                                  '' POINTER FOR PRIOR
              LET TRUK1 = NBR.OF.TRUCKS(CNV)  '' PERMITS PRINT TRUCKS
          WRITE TIME.V+TIME, TRUK1 , 1, DIR.CNV.CNTR(DIR) AS BINARY
                   USING DISK
'FIN'DESTROY THE SCHED.NEXT.CONVOY
      RETURN
      END
```

Appendix D1

THE INPUT TAPE FORMAT FOR THE PATTERN DETECTION ALGORITHM

If information from actual sensor data is to be input to the detection algorithm, the following format must be used:

Each logical record represents an activation and must be five words long sorted in ascending order on word 1.

Word 1--time of activation (decimal).
Word 2--sensor number of activation (integer).
Word 3--the integer 3.
Word 4--the integer 0.
Word 5--the integer 0.

The last three records of the file are special:

Record N-2 (same format)
Sensor 1--time of activation = time of last actual activation + 5.

Record N-1 (same format)
Sensor 1--time of activation = time of last actual activation + 10.

Record N (same format)
Word 3--the integer 9.
All other words, the value 0.

Appendix D2

THE OUTPUT FILE FORMAT CREATED BY THE SIMULATION MODEL

Each logical record is five words long and is sorted on the file in ascending order on word 1.

Word 1--time (decimal).

Word 2--sensor number if an activation, or number of trucks in the convoy if convoy has just entered the string (integer).

Word 3--type record code (integer):

1--convoy starts through the sensor field.

2--convoy leaves the sensor field.

3--false-alarm activation.

4--first vehicle of convoy passes the midpoint of the sensor.

5--last vehicle of convoy passes the midpoint of the sensor.

6--vehicle activation.

7--first vehicle of a convoy starts through a sensor.

8--last vehicle of a convoy leaves a sensor.

9--indicates last record of the file.

Word 4--convoy direction (integer 1 or 2).

Word 5--convoy number. Each convoy is numbered consecutively (1 to N) for *each* direction separately (integer).

The last three records are shown in Appendix D1.

Appendix E

INPUT DATA DECK FOR THE PATTERN DETECTION ALGORITHM

The following is a fully setup deck to run the pattern detection algorithm on the Rand IBM 360/65 computer installation. Tape number 002325 in this case contains the output activations of the simulation model, but it could contain actual activation data. (The format is described in Appendix D1 for actual data, and in Appendix D2 for simulation or experimental data.)

```
//C4300*06  JOB  (5772,300,120),'ANTHONY P. CIERVO',CLASS=A
//PAL120  EXEC SFORTG,NAME=LOGICX,LIB1='B4562.LIB2',REGION=196K
//GO.FT06F001  DD  SYSOUT=V,SPACE=(TRK,(950,1),RLSE),
//  DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)
//GO.FT08F001  DD UNIT=TAPE,DSN=B4562,VOL=SER=002325,
//  DCB=(RECFM=VB,BLKSIZE=2404,LRECL=24),
// DISP=OLD,LABEL=(,,,IN)
//GO.SYSIN  DD  *
  THIS IS AN EXAMPLE OF THE PATTERN DETECT. ALGOR. USING SIMULATED DATA  2/12/72
 1    24.0    35.0   15.0      24.0       35.0       15.0        0.40       0.05
       0.5  1000.0     .67               5          3           3          5   60.0  1
 0 1.0
 2    250.0      250.0      250.0      250.0
/*
//
```

Appendix F

THE SOURCE LISTING OF THE PATTERN DETECTION ALGORITHM

```
C MAIN ROUTINE - LOGIC BOX                          M. BERMAN 9 SEPT 71
C
C     THIS ROUTINE READS ALL NECESSARY INPUT INFORMATION. DIMENSIONS
C  ALL ARRAYS AND ZEROS THOSE ARRAYS. SOME PRELIMINARY CALCULATIONS ARE
C  MADE.
C                 *** DEFINITIONS ***
C
C AVGVEL(K)      - AVG. VELOCITY BETWEEN SENSORS FOR DIRECTION K. (KM/HR)
C                  (1 - WESTBOUND, 2 - EASTBOUND)
C
C BETA           - MAX TIME BETWEEN DETECTIONS IN A VALID STRIP. (MIN.)
C
C BWWND(K)       - MAX. VELOCITY FOR DIRECTION K.
C
C CSENS          - WEIGHT BELOW WHICH A SENSOR IS CONSIDERED HYPO-ACTIVE
C
C DIST(I)        - DISTANCE (M.) BETWEEN SENSOR I AND I - 1
C
C IB(I,K)        - POINTS TO THE NEXT AVAILABLE STORAGE CELL FOR SENSOR
C                  I, DIRECTION K
C
C IDROP(I)       -  MARKS A SENSOR I WHEN DROPPED FROM THE STRING
C                   ( 0 - NOT DROPPED, 1 - DROPPED)
C
C IKTRAJ(K)      -  COUNTS THE NBR. OF TRAJECTORIES STARTED, DIRECTION K
C
C INACTV(I)      -  THE NUMBER OF CONSECUTIVE PERIODS SENSOR I HAS BEEN
C                   HYPO-ACTIVE
C
C IWTIND         - COMPUTE WTS. INDICATOR. IF NOT 0 NONE WILL BE COMPUTD
C
C JSR(I)         - TEMP STORAGE FOR RNKING SENSOR NO. ON TIME
C
C KAGRAF         - USER GRAPH PLOTTING CONTROL. 0 - NO PLOT. 1 - GRAPH
C
C LASTSN(K)      - THE SENSOR NO. OF THE LAST SENSOR IN THE STRING FOR
C                  DIRECTION K.
C
C MSENS          - THE MINIMUM NUMBER OF ADMISSIBLE STRIPS FOR A TRAJ.
C                  CONFIRMATION
C
C NA(I)          - COUNTS THE NBR. OF ADMISSABLE STRIPS ON SENSOR I EACH
C                  PERIOD
C
C NASTC(L,K,I)   - ADMISSABLE STRIP FOR TRAJECTOR NO. L, DIREC. K  ON
C                  SENSOR I ?  1 - YES,  0 - NO
C
C NAVLID(I,J,K)  - INDICATOR WHICH SHOWS WHETHER OR NOT VALID STRIP WAS
C                  ADMISSABLE IN WINDOW OF CELL J , ON SENSOR I FOR
C                  DIRECTION K.
C
C NB             -  THE NO. OF TRAJECTORIES TO BE CONFIRMED BEFORE
C                   UPDATING WEIGHTS.
C
C NCARDR         -  THE DEVICE NO. OF THE CARD READER
C
```

```
C ND             - NO.OF CONSECUTIVE TIME PERIODS BEFORE HYPO-ACTIVE      MAIN 510
C                  SENSOR IS DROPPED FROM THE STRING                       MAIN 520
C                                                                          MAIN 530
C NDETEC(I)       -  COUNTS THE NO. OF DETECTIONS ON SENSOR I THAT ARE     MAIN 540
C                    SEPERATED BY LESS THAN BETA                           MAIN 550
C                                                                          MAIN 560
C NDISK           -  THE DEVICE NO. OF THE INPUT DATA DEVICE               MAIN 570
C                                                                          MAIN 580
C NDWSTR(L)       -  STORES THE DIRECTION OF THE L-TH CONFIRMED TRAJ       MAIN 590
C                                                                          MAIN 600
C NEND            -  THE NO. OF CELLS AVAILABLE IN THE WINDOW ARRAYS       MAIN 610
C                                                                          MAIN 620
C NOPENT(I,K)     - STORES THE CELL NO. OF THE FIRST OPEN WINDOW OF SENSORMAIN 630
C                   I FOR DIRECTION K.                                     MAIN 640
C                                                                          MAIN 650
C NSENSR          - INITAL NO. OF SENSORS IN THE STRING                    MAIN 660
C                                                                          MAIN 670
C NSNS            - THE MAXIMUM NUMBER OF SENSOR THE LOGIC BOX CAN HOLD    MAIN 680
C                                                                          MAIN 690
C NPRNT           -  THE DEVICE NO OF THE LINE PRINTER                     MAIN 700
C                                                                          MAIN 710
C NRTRAJ(I,J,K) - THE TRAJECTORY NO. OF THE WINDOW IN THE J-TH CELL ON     MAIN 720
C                 SENSOR I, DIRECTION K                                    MAIN 730
C                                                                          MAIN 740
C NTJSTR(L)       -  STORES THE TRAJ. NO. OF THE L-TH CONFIRMED TRAJ.      MAIN 750
C                                                                          MAIN 760
C NTRAJC          -  COUNTS THE NO. OF CONFIRMED TRAJECTORIES              MAIN 770
C                                                                          MAIN 780
C NV(I)           -  COUNTS VALID STRIPS ON SENSOR I EACH PERIOD           MAIN 790
C                                                                          MAIN 794
C PCTSEN           - THE PERCENT OF ACTIVE SENSORS REQ'D TO COMFIRM        MAIN 795
C                                                                          MAIN 800
C R(I)            -  COEFF. FOR SENSOR I EACH PERIOD                       MAIN 810
C                                                                          MAIN 820
C SEGLNT          -  ROAD SEGMENT LENGTH. (M.)                             MAIN 830
C                                                                          MAIN 840
C TIMFIN(I)       - STORED FINISH TIME OF A VALID STRIP ON SENSOR I.       MAIN 842
C                   STORED ONLY FOR THOSE SENSORS WHICH ARE MSENS OF THE   MAIN 843
C                   STRING END. (NOTE : ONLY ONE VALID STIP PER SENSOR IS  MAIN 844
C                 SAVED. A SUBSEQUENT STRIP WILL OVERLAY)                  MAIN 846
C                                                                          MAIN 847
C TIGUNO(I)       - STORED BEGIN TIME. (COMPLEMENT TO TIMFIN(I))           MAIN 848
C                                                                          MAIN 849
C TMEFST(I)       -  TIME OF FIRST IMPULSE IN A STRIP ON SENSOR I.         MAIN 850
C                                                                          MAIN 860
C TMELST(I)       -  TIME OF LAST IMPULSE IN A STRIP ON SENSOR I           MAIN 870
C                                                                          MAIN 872
C TMESR(I)        - TEMP STORAGE FOR RANKINK LAST DETEC TME FOR SENOR I    MAIN 873
C                                                                          MAIN 874
C TSBAR(I,K)      - AVG. TIME TO TRAVERSE DIST(I) FOR DIRECTION K.         MAIN 880
C                                                                          MAIN 881
C TVEL(I,K,M)     - TIME TO TRAVERSE DIST(I), DIRECTION K,FOR MIN VELOCITYMAIN 882
C                   (M = 1) OR MAX. VELOCITY (M = 2)                       MAIN 884
C                                                                          MAIN 890
C UPWND(K)        - MIN. VELOCITY FOR DIRECTION K.                         MAIN 900
C                                                                          MAIN 910
C W(I)            - THE WEIGHT    AN ADMISSABLE STRIP ON SENSOR I IS       MAIN 920
```

```
C                     CONTRIBUTING TO A CONFIRMATION EACH PERIOD          MAIN 930
C                                                                         MAIN 934
C WPRIME(I,N)     - THE SMOOTHED WEIGT OF SENSOR I DURING PERIOD N.       MAIN 936
C                                                                         MAIN 940
C WCAP            - THE SUM OF W(I) MUST BE GREATER THAN THIS FOR A TRAJ  MAIN 950
C                   CONFIRMATION                                          MAIN 960
C                                                                         MAIN 970
C WCNT            - MINIMUM NR OF DETECTIONS TO FORM A VALID STRIP        MAIN 980
C                                                                         MAIN 990
C WLWTM(I,J,K)    - WINDOW OPEN TIME OF WINDOW IN CELL J ON SENSOR I FOR  MAIN1000
C                   DIRECTION K.                                          MAIN1010
C                                                                         MAIN1020
C WUPTM(I,J,K)    - WINDOW CLOSE TIME OF WINDOW IN CELL J ON SENSOR I FOR MAIN1030
C                   DIRECTION K                                           MAIN1040
C                                                                         MAIN1050
C               +++ THE FOLLOWING ARE USED ONLY WHEN SIMULATING +++       MAIN1060
C                                                                         MAIN1070
C BEGTME(I,K)     -  THE TIME CONVOY I DIRECTION K ENTERS LAST SENSOR     MAIN1080
C                    OF THE STRING                                        MAIN1090
C                                                                         MAIN1100
C FINTME(I,K)     -  THE TIME CONVOY I DIRECTION K COMPLETES THE LAST     MAIN1110
C                    SENSOR OF THE STRING.                                MAIN1120
C                                                                         MAIN1130
C NFALSE          -  COUNTS THE NO. OF CONFIRMED TRAJ. NOT DUE TO CONVOYS MAIN1140
C                                                                         MAIN1150
C NRCGEN(N)       -  THE NO. OF CONVOYS GENERATED OF TRUCK SIZE N         MAIN1160
C                                                                         MAIN1170
C NRCNDT(N)       -  THE NO. OF CONVOYS DETECTED OF TRUCK SIZE  N         MAIN1180
C                                                                         MAIN1190
C NRTRUK(I,K)     -  THE NO. OF TRUCKS IN CONVOY I, DIRECTION K           MAIN1200
C                                                                         MAIN1210
C NSIZ3           - THE NO. OF CELLS AVAILABLE IN THE CONVOYS ARRAYS      MAIN1220
C                                                                         MAIN1230
C  ++++ SEE ROUTINE GRAPHG FOR VARIABLE DEFINITIONS OF GRAPHS. ++++       MAIN1232
C                                                                         MAIN1234
      COMMON/GRAPHS/ DISTR , GRMIN , GRPMAX, GRPMIN, IDIM1 , IDIM2   ,
     1         KAGRAF, KEAST , KEASTR, KFALSE, KTRKS , KWEST , KWESTR  ,
     A        MDIM3 , MDIM4 , MEAST , MEASTR, MFALSE, MTRKS , MWEST ,
     B        MWESTR, NOWIND, NWOUIT, KEAST1, MEAST1, KWEST1, MWEST1,
     2         Z(200), AFALSX(200), AFALSY(200), DIRPEX( 25), DIRPEY( 25),
     3                 DIRPWX( 25), DIRPWY( 25), EASTLX(150), EASTLY(150),
     4                 EASTUX(150), EASTUY(150), POINTX(200), POINTY(200),
     5                 WESTLX(150), WESTLY(150), WESTUX(150), WESTUY(150)
     6      ,TL(50),TAFLSX(100), TAFLSY(100), TORPEX( 15), TORPEY( 15),
     7               TORPWX( 15), TORPWY( 15), TESTLX( 75), TESTLY( 75),
     8               TESTUX( 75), TESTUY( 75), TPONTX(100), TPONTY(100),
     9               TWSTLX( 75), TWSTLY( 75), TWSTUX( 75), TWSTUY( 75)
      COMMON/XTRA/ TIMUNO(50), TIMFIN(50), WPRIME(15,150), PCTSEN, RHO
     1            , NA(100) ,JVALMN(100), JASTMN(100)
      DIMENSION AVGVEL(2)     ,COMENT(20)    ,DIST(15)        ,IDROP(15),    MAIN1240
     1          IKTRAJ(2)     ,INACTV(15)    ,LASTSN(2)       ,BWWND(2) ,    MAIN1250
     2          NAVLID(15,10,2),NASTC(40,2,15),NDRSTR(20) ,NDETEC(15),       MAIN1260
     3          NOPENT(15,2)  ,NRTRAJ(15,10,2),NTJSTR(20)   ,NV(15)     ,    MAIN1270
     4          R(15)         ,TMEFST(15)    ,TMELST(15)      ,TSHAR(16,2),  MAIN1280
     5          UPWND(2)      ,W(15,150)     ,WLWTM(15,10,2),                MAIN1290
     6          WUPTM(15,10,2), IH(15,2),    IERR(16), JSR(15),TMESR(15),MAIN1300
     7          TVEL(16,2,2)                                                  MAIN1304
```

```
C                                                                        MAIN1310
C**** THE FOLLOWING ARRAYS ARE ONLY FOR CONVOY STATISTICS AND ARE NOT    MAIN1320
C**** NECESSARY FOR OPERATION OF THE LOGIC BOX                           MAIN1330
C                                                                        MAIN1340
      DIMENSION BEGTME(50,2) ,FINTME(50,2) ,NRCGEN(20), NRCNOT(20),     MAIN1350
     1          NRTRUK(50,2) ,LSTCNV(2)                                  MAIN1360
C                                                                        MAIN1370
C *** CONSTANTS                                                          MAIN1380
C                                                                        MAIN1390
      IDIM1 = 200                                                        MAIN1394
      IDIM2 = 150                                                        MAIN1396
         MDIM3 = 100                                                     MAIN1397
         MDIM4 = 75                                                      MAIN1398
      NCARDR = 5                                                         MAIN1400
      NDISK = 8                                                          MAIN1410
      NPRNT = 6                                                          MAIN1420
      NEND  = 10                                                         MAIN1430
      NSIZ3 = 50                                                         MAIN1440
      NSNS  = 15                                                         MAIN1450
      NWARAY = 150                                                       MAIN1451
      NRTJC = 40                                                         MAIN1452
      NBRTRK = 20                                                        MAIN1453
C                                                                        MAIN1454
C**** ZERO ERROR ARRAY                                                   MAIN1455
C                                                                        MAIN1456
      DO 25 I = 1, 16                                                    MAIN1457
   25       IERR(I) = 0                                                  MAIN1458
C                                                                        MAIN1460
C**** READ THE COMMENT CARD - THEN PRINT THE COMMENT LINE                MAIN1470
C                                                                        MAIN1480
      READ(NCARDR,1) (COMENT(I), I = 1, 20)                              MAIN1490
      WRITE(NPRNT,2) (COMENT(I), I = 1, 20)                              MAIN1500
C                                                                        MAIN1510
C**** READ ALL INPUT CONTROL PARAMETERS AND PRINT THEM                   MAIN1520
C                                                                        MAIN1530
      READ(NCARDR,3)ID, (AVGVEL(I), BWWND(I), UPWND(I), I = 1,2), BETA,  MAIN1540
     1              CSENS, WCAP, SEGLNT,PCTSEN, NB, ND, IWCNT, NSENSR    MAIN1550
     2             , GRMIN , KAGRAF, NOWIND , RHO                        MAIN1552
           IF (ID .NE. 1) IERR(1) = 5                                    MAIN1560
           IF (AVGVEL(1) .LE. 0.0) IERR(2) = 10                          MAIN1570
           IF (AVGVEL(2) .LE. 0.0) IERR(3) = 15                          MAIN1580
           IF (BWWND(1)  .LE. 0.0) IERR(4) = 20                          MAIN1590
           IF (BWWND(2)  .LE. 0.0) IERR(5) = 25                          MAIN1600
           IF (UPWND(1)  .LE. 0.0) IERR(6) = 30                          MAIN1610
           IF (UPWND(2)  .LE. 0.0) IERR(7) = 35                          MAIN1620
           IF (BETA .LE. 0.0) IERR(8) = 40                               MAIN1630
           IF (CSENS .LE. 0.0) IERR(9) = 45                              MAIN1640
           IF (WCAP  .LE. 0.0) IERR(10) = 50                             MAIN1650
           IF (SEGLNT .LE. 0.0) IERR(11) = 55                            MAIN1660
           MSENS = PCTSEN * NSENSR                                       MAIN1664
           IF (MSENS  .LT. 2 )  IERR(12) = 60                            MAIN1670
           IF (NB   .LE. 0) IERR(13) = 65                                MAIN1680
           IF (ND   .LE. 0) IERR(14) = 70                                MAIN1690
           IF (IWCNT .LE. 0) IERR(15) = 75                               MAIN1700
           IF (NSENSR .GT. NSNS) IERR(15) = 72                           MAIN1704
      WRITE(NPRNT,4) (AVGVEL(I), BWWND(I), UPWND(I), I = 1,2), NSENSR,   MAIN1710
     1        SEGLNT ,PCTSEN, IWCNT, BETA, NB, CSENS, ND, WCAP           MAIN1720
```

```
C                                                                        MAIN1730
      WRITE(NPRNT, 10) KAGRAF , GRMIN, NOWIND                            MAIN1732
      WRITE(NPRNT, 11)  RHO                                              MAIN1736
C**** READ DISTANCE BETWEEN SENSORS & PRINT                              MAIN1740
C                                                                        MAIN1750
            DIST(1) = 0.0                                                MAIN1760
      READ(NCARDR,5) ID, (DIST(I), I = 2, NSENSR)                        MAIN1770
      WRITE(NPRNT,6)                                                     MAIN1780
                  TOT = 0.0                                              MAIN1790
            DO 20  I = 1, NSENSR                                         MAIN1800
                  TOT = TOT + DIST(I)                                    MAIN1810
   20       WRITE(NPRNT, 7)  I, DIST(I)                                  MAIN1820
                  IF(TOT .NE. SEGLNT) IERR(16) = 80                      MAIN1830
C                                                SET GRAPH PARAMETERS    MAIN1831
             KEAST  = 0                                                  MAIN1832
             KEASTR = 0                                                  MAIN1833
             KFALSE = 0                                                  MAIN1833
             KTRKS  = 0                                                  MAIN1833
             KWEST  = 0                                                  MAIN1833
             KWESTR = 0                                                  MAIN1833
                  KWEST1 = 0                                             MAIN1833
                  KEAST1 = 0                                             MAIN1833
                  MEAST1 = 0                                             MAIN1833
                  MWEST1 = 0                                             MAIN1833
                  MEAST  = 0                                             MAIN1833
                  MEASTR = 0                                             MAIN1833
                  MFALSE = 0                                             MAIN1833
                  MTRKS  = 0                                             MAIN1833
                  MWEST  = 0                                             MAIN1833
                  MWESTR = 0                                             MAIN1833
            NWQUIT = 0                                                   MAIN1833
             GRPMIN = 0.0                                                MAIN1834
             GRPMAX = GRMIN                                              MAIN1835
             DISTR  = NSENSR + 1                                         MAIN1836
C                                                                        MAIN1840
C**** SET AVG. TIME TO TRAVERSE EACH DISTANCE FOR EACH DIRECTION          MAIN1850
C                        (1) WESTBOUND                                   MAIN1860
C                                                                        MAIN1870
            TSBAR(1,1) = 0.0                                             MAIN1880
            DO 30  I = 2, NSENSR                                         MAIN1890
         TVEL(I,1,1) = DIST(I)/(UPWND(1)/60.0 * 1000.)                   MAIN1892
         TVEL(I,1,2) = DIST(I)/(BWWND(1)/60.0 * 1000.)                   MAIN1895
   30       TSBAR(I,1) = DIST(I)/(AVGVEL(1)/60.0 * 1000.)                MAIN1900
C                                                                        MAIN1910
C                        (2) EASTBOUND                                   MAIN1920
            N = NSENSR - 1                                               MAIN1930
            TSBAR(NSENSR,2) = 0.0                                        MAIN1940
            DO 40  I = 1, N                                              MAIN1950
         TVEL(I,2,1) = DIST(I+1)/(UPWND(2)/60.0 * 1000.)                 MAIN1952
         TVEL(I,2,2) = DIST(I+1)/(BWWND(2)/60.0 * 1000.)                 MAIN1954
   40       TSBAR (I,2) = DIST(I+1)/(AVGVEL(2)/60.0 * 1000.)             MAIN1960
C                                        SET LAST SENSORS                MAIN1964
            LASTSN(1) = NSENSR                                           MAIN1965
            LASTSN(2) = 1                                                MAIN1966
C                                                                        MAIN1970
C**** CHECK FOR INPUT ERROR                                              MAIN1980
C                                                                        MAIN1990
```

```
      IERROR = 0
      DO 100 I = 1, 16
  100          IERROR = IERROR + IERR(I)
      IF (IERROR) 200, 200, 110
  110 WRITE(NPRNT, 8)  (IERR(I), I = 1, 16)
      CALL EXIT
C
C**** SET UP DISTANCE OF X AXIS FOR GRAPHING
C
  200          TL(1) = .05
               TL(NSENSR) = .95
               ITEMP = NSENSR - 1
      DO 150  I = 2, ITEMP
  150          TL(I) = .9 * DIST(I)/SEGLNT + TL(I-1)
C                                                   SET UP GRAPH

      IF (KAGRAF .NE. 0) CALL GRAPH  (1, TIME, NPRNT)
C
C**** ZERO ALL ARRAYS
C
      DO 300  I = 1, NSENSR
               IDROP(I) = 0
               INACTV(I) = 0
               NDETEC(I) = 0
                  JSR(I) = 99999
                TMESR(I) = 999999.
               JASTMN(I) = 0
               JVALMN(I) = 0
                   NV(I) = 0
                   NA(I) = 0
                 W(I,1) = 1.0/ FLOAT(NSENSR)
            WPRIME(I,1) = W(I,1)
               TMEFST(I) = 0.0
               TMELST(I) = 0.0
                    R(I) = 0
               TIMUND(I) = 0.0
               TIMFIN(I) = 0.0
      DO 300  K = 1, 2
                  IB(I,K) = 1
               NOPENT(I,K) = 0
               IKTRAJ(K)   = 0
      DO 310  J = 1, NEND
               NAVLID(I,J,K) = 0
               NRTRAJ(I,J,K) = 0
               WLWTM(I,J, K) = 0.0
  310          WUPTM(I,J, K) = 0.0
      DO 300  J = 1, NRTJC
  300          NASTC(J,K, I) = 0
C
      DO 400 I = 1, NSIZ3
      DO 400 K = 1, 2
               LSTCNV(K) = 0
               NRTRUK(I,K) = 0
               BEGTME(I,K) = 0.0
  400          FINTME(I,K) = 0.0
      DO 500  I = 1, NBRTRK
               NRCGEN(I) = 0
```

```
  500       NRCNOT(I) = 0                                                MAIN2360
      WRITE(NPRNT, 9)                                                    MAIN2365
C                                                                        MAIN2370
C**** CALL THE FIRST ROUTINE                                              MAIN2380
C                                                                        MAIN2390
      CALL VALIDS(IDROP , IKTRAJ, INACTV, LASTSN, TVEL  , NAVILD, NASTC  MAIN2400
     1           , NDRSTR, NDETEC, NOPENT, NRTRAJ, NTJSTR, NV, R , TMEFSTMAIN2410
     2           , TMELST, TSHAR , UPWND , W     , WLWTM , WUPTM , IH    MAIN2420
     3           , BEGTME, FINTME, NRCGEN, NRCNOT, NRTRUK, NRTJC , NBRTRKMAIN2430
     4           , NEND  , NSIZ3 , NSNS  , BETA  , IWCNT , NSENSR, MSENS MAIN2440
     5           , CSENS , ND    , NB    , O     , LSTCNV,NDISK, NPRNT , MAIN2441
     6             JSR   , TMESR , WCAP  , NWARAY                    )   MAIN2442
      CALL EXIT                                                          MAIN2450
C                                                                        MAIN2460
C**** FORMATS                                                            MAIN2470
C                                                                        MAIN2480
    1 FORMAT(20A4)                                                       MAIN2490
    2 FORMAT(1H1,20X,57H*** PATTERN RECOGNITION FOR VEHICLE FLOW PAST SEMAIN2500
     1NSORS ***// 10X, 20A4)                                             MAIN2510
    3 FORMAT(I2,F8.0, 7F10.0/3F10.0,4I10,F8.2, I2/ I2 , F6.2)            MAIN2520
    4 FORMAT(1H0,28HWESTBOUND : AVG. VELOCITY = , F8.2,  7H KM/HR., 2X   MAIN2530
     1       21H  MAXIMUM VELOCITY = , F6.2,25H K/HR     MINIMUM VELOCITYMAIN2540
     2     , 3H = , F6.2//28H EASTBOUND: AVG. VELOCITY = , F8.2,         MAIN2550
     3       30H KM/HR.     MAXIMUM VELOCITY = , F6.2, 12H K/HR     MIN.  MAIN2560
     4       17HIMUM VELOCITY  = , F6.2, 5H K/HR//11H THERE ARE , I3,    MAIN2570
     5       30H SENSORS ON A ROAD SEGMENT OF ,F8.2, 14H M.  AT LEAST ,  MAIN2580
     6     F4.3,35H ARE NEEDED TO CONFIRM TRAJECTORIES// 9H A VALID ,    MAIN2590
     7       24HSTRIP CONTAINS AT LEAST , I3,24H DETECTIONS NO MORE THANMAIN2600
     8     ,F7.3,11H MIN. APART//1H , I3,24H TRAJ MUST BE CONFIRMED ,    MAIN2610
     9       55HBEFORE UPDATING WEIGHTS. ANY SENSOR HAVING A WT. BELOW ,MAIN2620
     A    F7.3,/  51H IS HYPO-ACTIVE AND WILL BE DROPPED FROM THE STRINGMAIN2630
     B        7H AFTER , I3, 21H CONSECUTIVE PERIODS.// 12H THE SUM OF ,MAIN2640
     C       29HWEIGHTS MUST BE GREATER THAN , F7.3,13H TO ACCEPT A ,    MAIN2650
     D       24HTRAJECTORY CONFIRMATION.)                                MAIN2660
    5 FORMAT(I2,F8.0, 7F10.0/(8F10.0))                                   MAIN2670
    6 FORMAT(1H0, 10X, 7H SENSOR, 5X,14H DISTANCE(M.) //)                MAIN2680
    7 FORMAT(1H0, 13X, I2, 11X, F7.2)                                    MAIN2690
    8 FORMAT(12H0*** ERRORS: , 16(2X, I3))                               MAIN2700
    9 FORMAT ( 1H1, 7X ,                                                 MAIN2701
     A       20H WINDOW CLOSED TIMES,52X,41HCONVOY ENTERS OR LEAVES (NO MAIN2701
     1TRUCK) STRING/1H ,5HSENSR, 3X, 5HCLOSE, 7X, 4HOPEN, 5X, 3HDIR, 2X,MAIN2702
     2     8HTRAJ NO., 39X, 10HCONVOY NO., 2X, 3HDIR,7X, 4HTIME,3X,      MAIN2706
     3     6HTRUCKS                                    )                 MAIN2708
   10 FORMAT(20HOGRAPHING CONTROL : , I2, 3X, 20HMINUTES PER GRAPH = ,   MAIN2710
     1       F8.2, 3X, 18HWINDOWS ON (=1) = , I4)                        MAIN2712
   11 FORMAT (29HOTHE SMOOTHING CONSTANT IS = ,F6.4)                     MAIN2714
      END                                                                MAIN2718
```

```
C ROUTINE CHKADM                            M. BERMAN  9/16/71          CHKA  10
C                                                                       CHKA  20
C   THIS ROUTINE CHECK A VALID STRIP ON SENSOR I FOR ADMISSABILITY. IN  CHKA  30
C  ADDITION IT WILL CLOSE ANY WINDOW THAT SHOULD BE CLOSED ON SENSOR I. CHKA  40
C                                                                       CHKA  50
      SUBROUTINE CHKADM (IDROP , IKTRAJ, INACTV, LASTSN, BWWND , NAVLID,CHKA  60
     1                   NASTC , NDRSTR, NPRNT , NOPENT, NRTRAJ, NTJSTR,CHKA  70
     2                   NV, R , FIRSTM, PASTTM, TSBAR , UPWND , W     ,CHKA  80
     3                   WLWTM , WUPTM , IB    , BEGTME, FINTME, NRGEN ,CHKA  90
     4                   NCNDT , NRTRUK, NRTJC , NBRTRK, NEND  , NSIZ3 ,CHKA 100
     5                   NSNS  , BETA  , IWCNT , NSENSR, MSENS , CSENS ,CHKA 110
     6                   ND, NB, I     , NTRAJC, LSTCNV, WCAP, NWARAY  )CHKA 120
C                                                                       CHKA 130
      COMMON/GRAPHS/ DISTR , GRMIN , GRPMAX, GRPMIN, IDIM1 , IDIM2   ,
     1        KAGRAF, KEAST , KEASTR, KFALSE, KTRKS , KWEST , KWESTR  ,
     A       MDIM3 , MDIM4 , MEAST , MEASTR, MFALSE, MTRKS , MWEST ,
     B       MWESTR, NOWIND, NWOUIT, KEAST1, MEAST1, KWEST1, MWEST1,
     2        Z(200), AFALSX(200), AFALSY(200), DIRPEX( 25), DIRPEY( 25),
     3                DIRPWX( 25), DIRPWY( 25), EASTLX(150), EASTLY(150),
     4                EASTUX(150), EASTUY(150), POINTX(200), POINTY(200),
     5                WESTLX(150), WESTLY(150), WESTUX(150), WESTUY(150)
     6      ,TL(50),TAFLSX(100), TAFLSY(100), TDRPEX( 15), TDRPEY( 15),
     7              TDRPWX( 15), TDRPWY( 15), TESTLX( 75), TESTLY( 75),
     8              TESTUX( 75), TESTUY( 75), TPONTX(100), TPONTY(100),
     9              TWSTLX( 75), TWSTLY( 75), TWSTUX( 75), TWSTUY( 75)
      COMMON/XTRA/ TIMUND(50), TIMFIN(50)
      DIMENSION        IDROP(1), NASTC(NRTJC,2,NSNS), NOPENT(NSNS,2),   CHKA 140
     1            IKTRAJ(1)       , WUPTM(NSNS,NEND,2 ), WLWTM(NSNS,NEND,2)CHKA 150
     2           ,NAVLID(NSNS,NEND,2), LASTSN(1)        ,NRTRAJ(NSNS,NEND,2)CHKA 160
     3    ,       IB(NSNS,2), INACTV(1), BWWND(1),NDRSTR(1), NTJSTR(1),  CHKA 164
     4            NV(1), R(1), TSBAR(1), UPWND(1), W(1), BEGTME(1),      CHKA 165
     5            FINTME(1), NRGEN(1), NCNDT(1), NRTRUK(1), LSTCNV(1)    CHKA 166
C                                                                       CHKA 170
C***  CHECK FOR ADMISSABILITY IN BOTH DIRECTIONS                        CHKA 180
C                                                                       CHKA 190
      DO 900  K = 1, 2                                                  CHKA 200
C                                                                       CHKA 210
C**** IS THERE A WINDOW OPEN ON THIS SENSOR ?                           CHKA 220
C                                                                       CHKA 230
      IF (NOPENT(I,K))1000, 90, 140                                     CHKA 240
C                                                                       CHKA 250
C**** START A NEW TRAJ. - OPEN A WINDOW IF REMAINING SENSOR CAN CONFIRM CHKA 260
C**** A TRAJECTORY.                                                     CHKA 270
C                                                   WESTBOUND ?         CHKA 280
   90 IF (K - 1 ) 120, 120, 95                                          CHKA 290
C                                                   ITS EASTBOUND       CHKA 300
   95 IF (I - MSENS) 106, 100, 100                                      CHKA 310
C                                              CHECK THE NO. REMOVED    CHKA 320
  100       ICHECK = I                                                  CHKA 330
      DO 105  I1 = 1, I                                                 CHKA 340
      IF (IDROP(I1)) 1010, 105, 103                                     CHKA 350
  103         ICHECK = ICHECK - 1                                       CHKA 360
  105 CONTINUE                                                          CHKA 370
      IF (ICHECK - MSENS) 106, 107, 107                                 CHKA 380
C                                                                       CHKA 380
C**** SAVE THE VALID STRIP THATS MSENS FROM THE STRING END.             CHKA 382
```

```
C                                                                       CHKA 383
  106 TIMUND(I) = FIRSTM                                                CHKA 384
      TIMFIN(I) = PASTTM                                                CHKA 385
      GO TO 900                                                         CHKA 386
C                                                                       CHKA 390
C*** START A NEW TRAJECTORY                                              CHKA 400
C                                                                       CHKA 410
  107             I1 = I - 1                                            CHKA 420
  110    IKTRAJ(K) = IKTRAJ(K) + 1                                      CHKA 430
C                                          CHECK SIZE OF NASTC ARRAY    CHKA 432
          ICKNAS  =((IKTRAJ(1) + IKTRAJ(2))/(NTRAJC + 1) + 1)           CHKA 434
      IF (ICKNAS  .GE.      NRTJC)  GO TO 1050                          CHKA 435
                  IK = MOD(IKTRAJ(K), NRTJC)                            CHKA 440
      IF (IK) 112 , 112, 115                                            CHKA 450
  112             IK = NRTJC                                            CHKA 460
C                                                 ZERO ADM.STRP CNTRCHKA 470
  115         DO 117  I2 = 1, NSENSR                                    CHKA 480
  117            NASTC(IK,K,I2) = 0                                     CHKA 490
C                                                 FIRST ADM. STRIP      CHKA 500
                 NASTC(IK,K,I)  = 1                                     CHKA 510
C                                          OPEN WINDOW ON NXT SENS      CHKA 520
C                                                                       CHKA 530
      CALL CHKOVL (I1, IK, K , I , FIRSTM, PASTTM, 0 , 0 , LASTSN,      CHKA 540
     1              IDROP , TSBAR , BWWND , UPWND , WLWTM , WUPTM ,     CHKA 550
     2              NOPENT, NAVLID, NRTRAJ, IB     , NEND  , NASTC ,    CHKA 560
     3         NPRNT, NRTJC , NSNS   , NSNS+1, NSENSR               )  CHKA 570
      GO TO 900                                                         CHKA 580
C                                                 WESTBOUND CHECKS      CHKA 590
  120            NSW = NSENSR + 1                                       CHKA 600
      IF (I - (NSW - MSENS)) 125, 125, 138                              CHKA 610
C                                          CHK THE NO REMOVED           CHKA 620
  125         ICHECK = NSW - I                                          CHKA 630
      DO 130  I1 = I, NSENSR                                            CHKA 640
      IF (IDROP(I1)) 1010, 130, 127                                     CHKA 650
  127         ICHECK = ICHECK - 1                                       CHKA 660
  130 CONTINUE                                                          CHKA 670
      IF (ICHECK - MSENS) 138, 135, 135                                 CHKA 680
C                                                                       CHKA 681
C*** DITTO REMARKS STATEMENT 106                                          CHKA 682
C                                                                       CHKA 683
  138 TIMUND(I) = FIRSTM                                                CHKA 684
      TIMFIN(I) = PASTTM                                                CHKA 685
      GO TO 900                                                         CHKA 686
  135             I1 = I + 1                                            CHKA 690
      GO TO 110                                                         CHKA 700
C                                                                       CHKA 710
C**** THERE IS AN OPEN WINDOW ON THIS SENSOR                              CHKA 720
C           OBTAIN CELL NO. OF FIRST OPEN WINDOW                        CHKA 730
C                                                                       CHKA 740
  140             J = NOPENT(I,K)                                       CHKA 750
C                                                                       CHKA 760
C**** DOES BOTTOM OF VALID STRIP EXCEED THIS WINDOW UPPER TIME ?          CHKA 770
C                                                                       CHKA 780
  142 IF (FIRSTM - WUPTM(I,J,K)) 144, 144, 200                          CHKA 790
C                                                                       CHKA 800
C**** NO! IS THE VALID STRIP BELOW THIS OPEN WINDOW ?                      CHKA 810
C                                                                       CHKA 820
```

```
  144 IF (PASTTM - WLWTM(I,J,K)) 90, 146,146                             CHKA 830
C                                                                        CHKA 840
C**** NO! ITS IN THE WINDOW. SHOULD WINDOW NOW BE CLOSED ?                CHKA 850
C                                                                        CHKA 860
C                                           COUNT THE ADM. STRIP         CHKA 862
  146 NASTC(NRTRAJ(I,J,K), K, I) = 1                                     CHKA 864
      IF (PASTTM - WUPTM(I,J,K)) 163, 148, 148                           CHKA 870
C                                                                        CHKA 880
C**** YES CLOSE IT. IS IT THE ONLY WINDOW OPEN ?                          CHKA 890
C                                                                        CHKA 900
  148 WRITE(NPRNT,1) I, WUPTM(I,J,K), WLWTM(I,J,K), K, NRTRAJ(I,J,K)     CHKA 910
C S-C 4060 OUTPUT HERE +++++++++                                         CHKA 920
      IF ( (KAGRAF .EQ. 0) .OR. (NOWIND .EQ. 0) ) GO TO 147              CHKA 932
            CALL  SETRAY ( K,I, WUPTM(I,J,K), WLWTM(I,J,K) )             CHKA 934
  147 IF (J - (IB(I,K) - 1)) 149 , 160, 1470                             CHKA 940
C                  NOTE: IB(IK) CAN BE SMALLER THAN J.                   CHKA 941
 1470 IF (J - (IB(I,K) + NEND - 1)) 149, 160, 1030                       CHKA 942
C                                                                        CHKA 950
C**** NO THERE ARE MORE WINDOWS OPEN. UPDATE THE WINDOW POINTER           CHKA 960
C                                                                        CHKA 970
  149         ITEMP   = J + 1                                            CHKA 980
      IF (ITEMP - NEND) 154, 154, 152                                    CHKA 990
  152         ITEMP   = 1                                                CHKA1000
       NOPENT(I,K) = ITEMP                                               CHKA1010
              162                                                        CHKA1020
C                                                                        CHKA1030
C**** THIS IS THE ONLY OPEN WINDOW. CLOSE IT.                             CHKA1040
                                                                         CHKA1050
  160  NOPENT(I,K) = 0                                                   CHKA1060
  162 NAVLID(I,J,K)= 0                                                   CHKA1070
C                                                                        CHKA1080
C**** IF ITS THE LAST SENSOR IN THE STRING THEN THE TRAJ. IS COMPLETE     CHKA1090
C                                                                        CHKA1100
      IF (I - LASTSN(K)) 165, 168, 165                                   CHKA1110
  168 CALL TRAJCM (NRTRAJ(I,J,K), J , K, PASTTM, IDROP , CSENS , INACTV,CHKA1120
     1             LASTSN, NASTC , NB, ND, NDRSTR, NTJSTR, MSENS , NV   ,CHKA1130
     2             NTRAJC, R      , TSHAR , WCAP  , W      , NSNS  , NSIZ3CHKA1140
     3           , NRTJC , NBRTRK, BEGTME, FINTME. NCNDT , NRTRUK. NPRNTCHKA1150
     4           , LSTCNV, NSNS+1, NSENSR, WLWTM(I,J,K),BWWND, NWARAY ) CHKA1160
      GO TO 900                                                          CHKA1170
C                                          MARK A VALID STRIP            CHKA1180
  163 NAVLID(I,J,K)= 1                                                   CHKA1190
C                                                                        CHKA1200
C                                    OPEN WINDOW ON NEXT SENSOR          CHKA1210
  165 IF (K - 2) 170, 180, 180                                           CHKA1220
  170              L = I +1                                              CHKA1230
                 GO TO 190                                               CHKA1240
  180              L = I - 1                                             CHKA1250
  190 CALL CHKOVL (L      , NRTRAJ(I,J,K), K , I , FIRSTM, PASTTM, 0 , 0,CHKA1260
     1             LASTSN, IDROP , TSHAR, BWWND , UPWND , WLWTM , WUPTM,CHKA1270
     2             NOPENT, NAVLID,NRTRAJ, IB     , NEND   , NASTC , NPRNT,CHKA1280
     3     NRTJC ,  NSNS   , NSNS+1,NSENSR                          )  CHKA1290
      GO TO 900                                                          CHKA1300
C                                                                        CHKA1310
C**** THERE IS MORE THAN ONE WINDOW OPEN OR WE MISSED OR BOTH.            CHKA1320
C                                                                        CHKA1330
C                                          CLOSE CURRENT WINDOW          CHKA1340
```

```
  200 WRITE(NPRNT,1) I, WUPTM(I,J,K), WLWTM(I,J,K), K, NRTRAJ(I,J,K)    CHKA1350
C  S-C 4060 OUTPUT HERE +++++++                                           CHKA1360
      IF ( (KAGRAF .EQ. 0) .OR. (NOWIND .EQ. 0) ) GO TO 203              CHKA1362
            CALL  SETRAY ( K,I, WUPTM(I,J,K), WLWTM(I,J,K) )              CHKA1364
C                                            IS THIS THE LAST SENSOR CHKA1370
  203 IF (I - LASTSN(K))207, 205, 207                                     CHKA1380
  205 CALL TRAJCM (NRTRAJ(I,J,K), J, K, WUPTM(I,J,K), IDROP , CSENS ,     CHKA1390
     1              INACTV, LASTSN, NASTC , NB, ND, NDRSTR, NTJSTR ,MSENSCHKA1400
     2            , NV    , NTRAJC, R     , TSBAR , WCAP  , W     ,NSNS CHKA1410
     3            , NSIZ3 , NRTJC , NBRTRK, BEGTME, FINTME, NCNDT,VRTRUK CHKA1420
     4            , NPRNT , LSTCNV, NSNS+1, NSENSR, WLWTM(I,J,K), BWWND, CHKA1430
     5              NWARAY                                             ) CHKA1432
      GO TO 225                                                          CHKA1440
C                                                                        CHKA1450
C**** HAD A VALID STRIP EVER FELL IN THAT WINDOW ?                        CHKA1460
C                                                                        CHKA1470
  207 IF (NAVLID(I,J,K)) 1040, 210, 220                                  CHKA1480
C                                                                        CHKA1490
C**** NO! EXTEND TRAJECTORY                                              CHKA1500
C                                                                        CHKA1510
  210 IF (K -2) 214, 215, 215                                            CHKA1520
  214              L = I + 1                                             CHKA1530
                 GO TO 216                                               CHKA1540
  215              L = I - 1                                             CHKA1550
  216 CALL CHKOVL (L     , NRTRAJ(I,J,K) , K , I , 0.0, 0.0, 1 , J     , CHKA1560
     1              LASTSN, IDROP , TSBAR , BWWND , UPWND , WLWTM, WUPTM,CHKA1570
     2              NOPENT, NAVLID, NRTRAJ, IB    , NEND  , NASTC, NPRNT,CHKA1580
     3     NRTJC,   NSNS  , NSNS+1, NSENSR                          )    CHKA1590
      GO TO 225                                                          CHKA1600
C                                                                        CHKA1610
C**** A VALID STRIP HAS FALLEN IN THIS WINDOW.  ZERO CNTR                CHKA1620
C                                                                        CHKA1630
  220 NAVLID(I,J,K) = 0                                                  CHKA1640
C                                                                        CHKA1650
C**** LOOK AT THE NEXT WINDOW IF THERE IS ONE.                           CHKA1660
C                                                                        CHKA1670
  225              J = J + 1                                             CHKA1680
      IF (J - NEND) 228, 228, 226                                        CHKA1690
  226              J = 1                                                 CHKA1700
  228 IF (J - IB(I,K)) 230, 240, 230                                     CHKA1710
C                                            THERE IS. UPDATE PNTR       CHKA1720
  230   NOPENT(I,K) = J                                                  CHKA1730
      GO TO 142                                                          CHKA1740
C                                            NO MORE WINDOWS             CHKA1750
  240   NOPENT(I,K) = 0                                                  CHKA1760
      GO TO 40                                                           CHKA1770
  900 CONTINUE                                                           CHKA1780
      RETURN                                                             CHKA1790
C                                                                        CHKA1800
C**** FORMATS AND MESSAGES                                               CHKA1810
C                                                                        CHKA1820
    1 FORMAT(1H , I2, 3X, F9.3, 3X, F9.3, 4X, I2, 4X, I3)                CHKA1830
 1000 WRITE(NPRNT, 1001) NOPENT(I,K), I, K                               CHKA1840
 1001 FORMAT (15H0*** ERROR 500 , 3I10)                                  CHKA1850
      CALL EXIT                                                          CHKA1860
 1010 WRITE(NPRNT,1011) IDROP(I) , I , K                                 CHKA1870
 1011 FORMAT(15H0*** ERROR 510 , 3I10)                                   CHKA1880
```

```
      CALL EXIT                                                         CHKA1890
 1030 WRITE(NPRNT,1031)  J,  IB(I,K), I, K                              CHKA1900
 1031 FORMAT(15HO*** ERROR 520 , 4I10)                                  CHKA1910
      CALL EXIT                                                         CHKA1920
 1040 WRITE(NPRNT,1041)  NAVLID(I,J,K), I, J, K                         CHKA1930
 1041 FORMAT(15HO*** ERROR 530, 4I10)                                   CHKA1940
      CALL EXIT                                                         CHKA1950
 1050      ICKNAS  =(ICKNAS  + 1)/2                                     CHKA1952
      WRITE(NPRNT,1051) ICKNAS , IKTRAJ(1), IKTRAJ(2)                   CHKA1954
 1051 FORMAT(51HO***** ERROR 527: ARRAY NASTC(NRTJC, 2, NSNS) MUST ,    CHKA1956
     1             47HHAVE NRTJC DIMENSION AND INDEX INCREASED ABOVE ,I5,CHKA1957
     2          // 1HO, 16X,12HTRAJ. CMP = , I4, 19HTRAJ BEGUN, WEST = ,CHKA1958
     3              I4,  8H EAST = , I4)                                CHKA1959
      CALL EXIT                                                         CHKA195A
      END                                                               CHKA1960
```

```
C ROUTINE CHKOVL                                   M. BERMAN  9/13/71
C
C      THIS ROUTINE IS CALLED WHEN A WINDOW IS CLOSED ON A SENSOR. IT
C    OPENS THE WINDOW ON THE NEXT SENSOR. IT CHECK FOR WINDOW OVERLAP.
C    THAT IS IF A WINDOW TO BE OPENED OVERLAPS A PREVIOUS ONE THE A
C    SINGLE LARGE WINDOW IS MADE. IF THE OVERLAP IS FROM TWO DIFFERENT
C    TRAJECTORIES ONE LARGE WINDOW IS CREATED **WITH THE TRAJECTOR NO.
C    OF THE LOWER WINDOW. THE ROUTINE ALSO EXTENDS WINDOWS. THAT IS, IF
C    A WINDOW IS CLOSED THAT NEVER CONTAINED A VALID STRIP IT IS EXTENDED
C    THE AVERAGE TIME. OVERLAP IS ALSO CHECKED DURING EXTENSION.
C
      SUBROUTINE CHKOVL(L, NTJ, K , I6, TMEFT,  TMELT,  IEXND, NRCEL ,
     1                   LASTSN, IDROP , TSBAR, TVEL   , UPWND , WLWTM ,
     2                   WUPTM , NOPENT, NAVLID,NRTRAJ, IB    , NEND  ,
     3                   NASTC , NPRNT , NRTJC ,NSNS,   NSN2  , NSENSR )
C
C       L     - SENSOR NO ON WHICH WINDOW IS TO BE OPENED
C       NTJ   - TRAJECTORY NUMBER
C       K     - DIRECTION
C       I     - PREVIOUS SENSOR
C       NRCEL - CELL NO OF WINDOW ON PREVIOUS SENSOR THAT IS CLOSED
C
      COMMON/XTRA/ TIMUND(50), TIMFIN(50)
      DIMENSION LASTSN(1), IDROP(1),          UPWND(1), IB(NSNS,2) ,
     1            NASTC(NRTJC,2,NSNS),TSBAR(NSN2,2), WLWTM(NSNS,NEND,2),
     2            WUPTM(NSNS,NEND,2) ,NOPENT(NSNS,2),NAVLID(NSNS,NEND,2),
     3            NRTRAJ(NSNS,NEND,2), TVEL(NSN2,2,2)
        I = I6
        TMEFST = TMEFT
        TMELST = TMELT
        IEXTND = IEXND
C
C**** WAS PREVIOUS SENSOR LAST IN THE STRING
C
   80 IF (K - 2) 85, 90, 1010
   85 IF (I - LASTSN(K)) 100, 95,95
   90 IF (I - LASTSN(K)) 95, 95, 100
   95 RETURN
C
C**** NO! SEE IF SENSOR L HAS BEEN REMOVED.
C
  100 IF (IDROP(L)) 1000  200, 110
C
C**** IT HAS SKIP A SENSOR
C
  110 IF (K - 2 ) 120,130,1010
  120       L = L + 1
            GO TO 100
  130       L = L - 1
            GO TO 100
C
C**** I. CALCULATE THE NEW WINDOW UP AND DOWN TIMES.
C
C**** IS IT AN EXTENSION OF A WINDOW ?
C
  200 IF (IEXTND) 1020, 210, 220
C
```

```
C**** NO! ITS CAUSED BY AN ADMISSABLE STRIP                               COVL 510
C                                                                         COVL 520
  210 WOTME = (TMELST + TMEFST) * .5 + TVEL(L,K,2)                        COVL 530
      WCTME = (TMELST + TMEFST) * .5 + TVEL(L,K,1)                        COVL 540
      GO TO 300                                                           COVL 550
C                                                                         COVL 560
C**** YES! EXTEND WINDOW                                                  COVL 570
C                                                                         COVL 580
  220 WOTME = WLWTM(I, NRCEL, K) + TSBAR(L,K)                             COVL 590
      WCTME = WUPTM(I, NRCEL, K) + TSBAR(L,K)                             COVL 600
C                                                                         COVL 610
C**** II. CHECK FOR OVERLAP IF THERE IS AN OPEN WINDOW ON THIS SENSOR     COVL 620
C                                                                         COVL 630
  300 IF(NOPENT(L,K)) 1030, 310, 400                                      COVL 640
C                                                                         COVL 650
C****       (A) NO OPEN WINDOW. OPEN ONE.                                 COVL 660
C                                                                         COVL 670
  310        N = IB(L,K)                                                  COVL 680
             NOPENT(L,K)   = N                                            COVL 690
  320         WUPTM(L,N,K) = WCTME                                        COVL 700
              WLWTM(L,N,K) = WOTME                                        COVL 710
C                                                  ZERO VALID STRIP CNTR. COVL 720
             NAVLID(L,N,K) = 0                                            COVL 730
C                                                  SET TRAJECTORY NO.     COVL 740
             NRTRAJ(L,N,K) = NTJ                                          COVL 750
C                                                  SAVE THIS CELL NO.     COVL 754
                       NS3 = N                                            COVL 756
C                                                  UPDATE FIRST AVAIL CELL COVL 760
                         N = N + 1                                        COVL 770
        IF (N .GT. NEND) N = 1                                            COVL 780
                   IB(L,K) = N                                            COVL 790
C                                                                         COVL 791
C**** THIS SECTION TESTS TO SEE IF THIS NEW WINDOW NOW BRACKETS A         COVL 792
C**** PREVIOUS VALID STRIP THAT WAS TOO CLOSE TO THE STRING END TO BE     COVL 793
C**** ADMISSIBLE. IT MAY NOW CAUSE AN ASTI AN OPEN A WINDOW ON NEXT SEN.COVL 795
C                                                                         COVL 796
      IF (TIMFIN(L) - WOTME) 390, 330, 330                                COVL 797
  330 IF(TIMUNO(L) - WCTME) 340, 340, 390                                 COVL 798
C                                           ITS IN NEW WINDOW             COVL 799
  340 NAVLID(L,NS3,K) = 1                                                 COVL 79A
      NASTC(NTJ,K,L)= 1                                                   COVL 79B
C**** OPEN WINDOW ON NEXT SENSOR                                          COVL 79C
      TMELST = TIMFIN(L)                                                  COVL 79D
      TMEFST = TIMUNO(L)                                                  COVL 79E
      IEXTND = 0                                                          COVL 79F
      I = L                                                               COVL 79G
      IF (K - 2) 350,360, 1010                                            COVL 79H
  350 L = L + 1                                                           COVL 79I
      GO TO 80                                                            COVL 79J
  360 L = L - 1                                                           COVL 79K
      GO TO 80                                                            COVL 79L
  390 RETURN                                                              COVL 800
C                                                                         COVL 810
C****       (B) THERE IS A WINDOW OPEN                                    COVL 820
C                                                                         COVL 830
C                                                OBTAIN LAST OPEN CELL NOCOVL 840
  400                  J1 = IB(L,K) - 1                                   COVL 850
```

```
      IF (J1 .LT. 1) J1 = NEND                                          COVL 860
C                                               OBTAIN ITS TRAJECTORY NOCOVL 870
                  J2 = NRTRAJ(L, J1, K)                                 COVL 880
C                                                                       COVL 890
C****             IS NEW WINDOW FROM SAME TRAJECTORY AS THE OLD         COVL 900
C                                                                       COVL 910
         IF (J2 - NTJ) 420, 410, 420                                    COVL 920
C                                                                       COVL 930
C****             YES! JUST EXTEND UPPER WINDOW                         COVL 940
C                                                                       COVL 950
  410    WUPTM(L,J1,K) = WCTME                                          COVL 960
            RETURN                                                      COVL 964
C                                                                       COVL 970
C****             NO! CHECK FOR OVERLAP.                                COVL 980
C                                                                       COVL 990
  420    IF (WOTME - WUPTM(L,J1,K)) 500, 500,430                        COVL1000
C                                                                       COVL1010
C****                     NO OVERLAP. OPEN A NEW WINDOW                 COVL1020
C                                                                       COVL1030
  430             N = IB(L,K)                                           COVL1040
                  GO TO 320                                             COVL1050
C                                                                       COVL1060
C****     (C) WINDOWS OVERLAP. EXTEND CURRENT WINDOW                    COVL1070
C                                                                       COVL1080
  500    WUPTM(L,J1,K) = WCTME                                          COVL1090
C                                                                       COVL1100
C                                                                       COVL1110
C****             MERGE ADMISSABLE STRIPS INTO CURRENT WINDOW.          COVL1120
C                                                                       COVL1130
      DO 520  I2 = 1, NSENSR                                            COVL1140
         IF (NASTC(NTJ, K, I2)) 1040, 520, 510                          COVL1150
  510          NASTC(J2,  K, I2) = 1                                    COVL1160
  520 CONTINUE                                                          COVL1170
      RETURN                                                            COVL1180
C                                                                       COVL1190
C**** ERROR MESSAGES                                                    COVL1200
C                                                                       COVL1210
 1000 WRITE(NPRNT,1001)  L , IDROP(L)                                   COVL1220
 1001 FORMAT(15HO*** ERROR 230 , 2I10)                                  COVL1230
      CALL EXIT                                                         COVL1240
 1010 WRITE(NPRNT,1011)  K                                              COVL1250
 1011 FORMAT(15HO*** ERROR 240 , I10)                                   COVL1260
      CALL EXIT                                                         COVL1270
 1020 WRITE(NPRNT,1021) IEXTND                                          COVL1280
 1021 FORMAT(15HO*** ERROR 250 , I10)                                   COVL1290
      CALL EXIT                                                         COVL1300
 1030 WRITE(NPRNT, 1031) L, K,NOPENT(L,K)                               COVL1310
 1031 FORMAT(15HO*** ERROR 260 , 3I10)                                  COVL1320
      CALL EXIT                                                         COVL1330
 1040 WRITE(NPRNT, 1041) NTJ, K, I2, NASTC(NTJ,K, I2)                   COVL1340
 1041 FORMAT(15HO*** ERROR 270 , 4I10)                                  COVL1350
      CALL EXIT                                                         COVL1360
      END                                                               COVL1370
```

```
C ROUTINE CHKWIN                                     M. BERMAN  9/13/71      CWIN  10
C                                                                            CWIN  20
C      THIS ROUTINE CHECKS WINDOW IN BOTH DIRECTIONS TO  SEE IF ANY          CWIN  30
C    SHOULD BE CLOSED. THE CHECKS ARE MADE IN THE PROPER SEQUENTIAL ORDERCWIN  40
C    FOR EACH DIRECTION . THE ASSURES THAT WINDOW WILL BE CLOSED IN THE  CWIN  50
C    PROPER ORDER.                                                           CWIN  60
C      THE ROUTINE OPERATES IN TWO MODES:                                    CWIN  70
C           (1) ALL WINDOWS ARE CHECKED - NO VALID STRIP                     CWIN  80
C           (2) ALL WINDOWS EXCEPT FOR SENSOR I ARE CHECKED - VALID STRIPCWIN  90
C                                                          ON SENSOR ICWIN 100
C                                                                            CWIN 110
      SUBROUTINE CHKWIN(IDROP , IKTRAJ, INACTV, LASTSN, BWWND , NAVLID, CWIN 120
     1                  NASTC , NDRSTR, NPRNT , NOPENT, NRTRAJ, NTJSTR, CWIN 130
     2                  NV, R , FSTTME, PSTTME, TSBAR , UPWND , W     , CWIN 140
     3                  WLWTM , WUPTM , IB    , BEGTME, FINTME, NRGEN , CWIN 150
     4                  NCNDT , NRTRUK, NRTJC , NBRTRK, NEND  , NSIZ3 , CWIN 160
     5                  NSNS  , BETA  , IWCNT , NSENSR, MSENS , CSENS , CWIN 170
     6                  ND    , NB    , IS    , ISWTCH, NTRAJC, LSTCNV, CWIN 180
     7                  WCAP , TMELST, TMEFST, NWARAY              )    CWIN 180
C                                                                        CWIN 190
      COMMON/GRAPHS/ DISTR , GRMIN , GRPMAX, GRPMIN, IDIM1 , IDIM2   ,
     1        KAGRAF, KEAST , KEASTR, KFALSE, KTRKS , KWEST , KWESTR  ,
     A       MDIM3 , MDIM4 , MEAST , MEASTR, MFALSE, MTRKS , MWEST ,
     B       MWESTR, NOWIND, NWOUIT, KEAST1, MEAST1, KWEST1, MWEST1,
     2        Z(200), AFALSX(200), AFALSY(200), DIRPEX( 25), DIRPEY( 25),
     3               DIRPWX( 25), DIRPWY( 25), EASTLX(150), EASTLY(150),
     4               EASTUX(150), EASTUY(150), POINTX(200), POINTY(200),
     5               WESTLX(150), WESTLY(150), WESTUX(150), WESTUY(150)
     6      ,TL(50),TAFLSX(100), TAFLSY(100), TORPEX( 15), TDRPEY( 15),
     7             TORPWX( 15), TORPWY( 15), TESTLX( 75), TESTLY( 75),
     8             TESTUX( 75), TESTUY( 75), TPONTX(100), TPONTY(100),
     9             TWSTLX( 75), TWSTLY( 75), TWSTUX( 75), TWSTUY( 75)
      DIMENSION IB(NSNS,2), IDROP(1), NAVLID(NSNS,NEND,2),NOPENT(NSNS,2)CWIN 200
     1        , LASTSN(1) , WUPTM(NSNS, NEND, 2) , WLWTM(NSNS,NEND,2)   CWIN 210
     2        , NRTRAJ(NSNS,NEND,2), IKTRAJ(1), INACTV(1), BWWND(1)  ,  CWIN 212
     3          NASTC(1), NDRSTR(1), NTJSTR(1), NV(1), R(1), TSBAR(1),  CWIN 213
     4          UPWND(1), W(1), BEGTME(1), FINTME(1), NRGEN(1),NCNDT(1),CWIN 214
     5          LSTCNV(1), NRTRUK(1), TMELST(1), TMEFST(1)              CWIN 215
C                                                                        CWIN 220
            NSS = NSENSR + 1                                             CWIN 230
C                                                                        CWIN 240
C**** START WESTBOUND DIRECTION FIRST                                    CWIN 250
C                                                                        CWIN 260
      DO 900 K = 1, 2                                                    CWIN 270
            I = 0                                                        CWIN 280
         DO 910  I1 = 1, NSENSR                                          CWIN 290
            IF (K - 2) 100, 110, 110                                     CWIN 300
  100       I = I + 1                                                    CWIN 310
            GO TO 120                                                    CWIN 320
  110       I = NSS - I1                                                 CWIN 330
C                                                                        CWIN 340
C**** CONTINUE IF SENSOR IS DROPPED FROM THE STRING                      CWIN 350
C                                                                        CWIN 360
  120       IF (IDROP(I)) 1000, 130, 910                                 CWIN 370
C                                                                        CWIN 380
C**** CONTINUE IF NO WINDOW IS OPEN ON THE SENSOR                        CWIN 390
```

```
C
  130        IF (NOPENT(I, K)) 1010, 910, 140
C
C**** CONTINUE IF THE STRIP HAS VALID ON THIS SENSOR
C
  140        IF (ISWTCH) 160,160, 150
  150        IF (I - IS) 160,910, 160
C
C**** OBTAIN CELL NO. OF FIRST OPEN WINDOW ON THIS SENSOR
C
  160        J = NOPENT(I,K)
C
C**** SHOULD WINDOW BE CLOSED. IF NOT CONTINUE
C
  170        IF (PSTTME - WUPTM(I,J,K)) 910, 910, 172
C                                                  CHK FOR EXTENDED
  172         IF (TMEFST(I) -  WUPTM(I,J,K)) 174, 174, 180
  174         IF (PSTTME - TMFLST(I) - BETA) 910, 910,180
C
C**** CLOSE WINDOW SECTION
C
  180 WRITE(NPRNT, 1) I, WUPTM (I,J,K) , WLWTM(I,J,K), K , NRTRAJ(I,J,K)
C S-C 4060 OUTPUT HERE
      IF ( (KAGRAF .EQ. 0) .OR. (NOWIND .EQ. 0) ) GO TO 188
             CALL  SETRAY ( K, I, WUPTM(I,J,K), WLWTM(I,J,K) )
C
C**** SEE IF A TRAJECTORY IS COMPLETED
C
  188        IF (LASTSN(K) - I) 200, 190, 200
C
C**** TRAJECTORY COMPLETED
C
  190 CALL TRAJCM (NRTRAJ(I, J,K) , J , K , WUPTM(I,J,K), IDROP , CSENS
     1             , INACTV, LASTSN , NASTC , NB, ND, NDRSTR , NTJSTR,
     2               MSENS , NV      , NTRAJC, R       , TSBAR   , WCAP, W,
     3               NSNS   , NSIZ3   , NRTJC  , NBRTRK, BEGTME , FINTME ,
     4               NCNDT  , NRTRUK, NPRNT, LSTCNV, NSNS+1, NSENSR ,
     5               WLWTM(I,J,K)    , BWWND , NWARAY                     )
             GO TO 220
C
C**** HAS A VALID STRIP EVER FALLEN IN THE WINDOW ?
C
  200        IF (NAVLID(I, J, K)) 1020, 210, 220
C
C**** NO VALID STRIPS. EXTEND WINDOW.
C
  210        I3 = I + 1
             IF (K .EQ. 2) I3 = I - 1
      CALL CHKOVL(I3,NRTRAJ(I,J,K), K , I, FSTTME, PSTTME, 1, J, LASTSN,
     1              IDROP, TSBAR, BWWND, UPWND, WLWTM , WUPTM, NOPENT,
     2              NAVLID,NRTRAJ, IB   , NEND , NASTC , NPRNT ,NRTJC,
     3              NSNS    , NSNS+1, NSENSR                       )
C
C**** IT HAS HAS VALID STRIPS GO TO NEXT OPEN WINDOW IF THERE IS ONE
C
  220        J = J + 1
             IF (J .GT. NEND) J = 1
```

```
            IF (J - IB(I,K)) 230, 240, 230
C
C**** MORE OPEN WINDOWS. UPDATE OPEN WINDOW POINTER
C
  230       NOPENT(I,K) = J
            GO TO 170
C
C**** NO MORE OPEN WINDOWS ON SENSOR I.
C
  240       NOPENT(I,K) = 0
  910    CONTINUE
  900 CONTINUE
      RETURN
C
C**** ERROR MESSAGES
C
 1000 WRITE(NPRNT,10) I ,IDROP(I)
   10 FORMAT(15H0*** ERROR 210 ,2I10)
      CALL EXIT
 1010 WRITE(NPRNT,20) I, K, NOPENT(I,K)
   20 FORMAT(15H0*** ERROR 220, 3I10)
      CALL EXIT
 1020 WRITE(NPRNT,1021) I, J, K, NAVLID(I,J,K)
 1021 FORMAT(15H0*** ERROR 230 ,4I10)
      CALL EXIT
    1 FORMAT( 1H , I2, 3X, F9.3, 3X, F9.3, 4X, I2, 4X, I3)
      END
```

```
C ROUTINE CNTTRJ                              M. BERMAN  9/15/71
C
C      THIS ROUTINE IS CALLED WHEN A TRAJECTORY IS CONFIRMED. IT
C    DETERMINES IF THE CONFIRMED TRAJECTORY IS IN FACT A CONVOY
C    TRAJECTORY
C
      SUBROUTINE CNTTRJ (TMCLOS, TMOPEN, K, BEGTME, FINTME, NRTRUK,
     1                   NRCNDT, NSIZ3 , LSTCNV                       )
C
      DIMENSION BEGTME(NSIZ3,2), FINTME(NSIZ3,2), NRTRUK(NSIZ3, 2),
     1          NRCNDT(1), LSTCNV(1)
C                                             OBTAIN PNTR TO LAST CONVOY
          N = LSTCNV(K)
C                                          IF 0 A DETECTION WAS MADE PRIOR
      IF (N) 400, 400, 90
C                                             IS WINDOW BELOW CONVOY STRIP?
   90 IF (TMCLOS  -   BEGTME(N,K)) 200, 100, 100
C                                             NO. IS CONVOY STRIP COMPLETED?
C
  100 IF (BEGTME(N,K) - FINTME(N,K)) 110, 400 , 150
C
C                                             IS WINDOW ABOVE CONVOY STRIP
  110 IF (TMOPEN - FINTME(N,K)) 150, 150, 400
C
C**** CONVOY IS DETECTED
C
  150 NRCNDT(NRTRUK(N,K)) = NRCNDT(NRTRUK(N,K)) + 1
C                                          PERMIT ONLY 1 DETECN/CONVOY
      LSTCNV(K) = 0
      RETURN
C
C**** GO TO EXAMINE THE PREVIOUS CONVOY STRIP
C
  200       N = N - 1
      IF (N)220, 210, 220
  210       N = NSIZ3
  220 GO TO 90
                                              ITS A FALSE TRAJECTORY
  400 RETURN
      DEBUG INIT (NRCNDT)
      END
```

```
C SUBROUTINE GRAPH                                M. BERMAN 11/11/71
C
C  THIS ROUTINE IS CALLED (I) WHEN SETTING UP INITAL GRAPH. (II) WHEN
C ANY OF THE COORDINATE ARRAYS FOR DETECTIONS OR FALSE ALARMS ARE
C FILLED. (III) WHEN AN IMPULSE IS LARGER IN TIME THAN THE CURRENT
C GRAPHS Y(MAX).
C
C                 ++++ DEFINITIONS FOR GRAPHING +++
C
C AFALSX(IDIM1) - SENSOR DISTANCE COORDINATE FOR A FALSE ALARM
C
C AFALSY(IDIM1) - TIME COORDINATE FOR A FALSE ALARM
C
C DIRPEX(IDIM3) - X COORDINATE OF TRAJ. COMPLETE MARKER. EASTBOUND TRAJ.
C
C DIRPEY(IDIM3) - TIME COORDINATE OF TRAJ. COMPLETE MARK. EASTBND TRAJ.
C
C DIRPWX(IDIM3) - X COORDINATE OF TRAJ. COMPLETE MARKER. WESTBOUND TRAJ.
C
C DIRPWY(IDIM3) - TIME COORDINATE OF TRAJ. COMPLETE MARK. WESTBND TRAJ.
C
C DISTR         - THE MAX. X COORDINATE. NSENSR + 1 .
C
C EASTLX(IDIM2) - SENSOR COORD. FOR LOWER WINDOW OF EASTBND TRAJ.
C
C EASTLY(IDIM2) - TIME COORD. FOR LOWER WINDOW OF EASTBOUND TRAJ.
C
C EASTUX(IDIM2) - SENSOR COORD. FOR UPPER WINDOW OF EASTBOUND TRAJ
C
C EASTUY(IDIM2) - TIME COORD. FOR UPPER WINDOW OF EASTBOUND TRAJ.
C
C GRMIN         - THE USER SPECIFIED MINUTES PER GRAPH.
C
C GRPMIN        - THE MINIMUM TIME COORD FOR A GRAPH
C
C GRPMAX        - THE MAXIMUM TIME COORD FOR A GRAPH
C
C IDIM1         - THE DIMENSION OF THE AFALS AND POINT ARRAYS. |SET IN
C                                                              |THE MAIN
C IDIM2         - THE DIMENSION OF THE EAST  AND WEST ARRAYS.  |RTN.
C
C KAGRAF        - USER GRAPH PLOTTING CONTROL. 0 - NO PLOT. 1 - PLOT
C
C KEAST         - COUNTS THE ENTRIES IN THE EAST ARRAYS.
C
C KEASTR        - COUNTS THE ENTRIES IN THE DIRPE ARRAYS.
C
C KFALSE        - COUNTS THE ENTRIES IN THE AFALS ARRAYS
C
C KTRKS         - COUNTS THE ENTRIES IN THE POINT ARRAYS.
C
C KWEST         - COUNTS THE ENTRIES IN THE WEST ARRAYS
C
C KWESTR        - COUNTS THE ENTRIES IN THE DIRPW ARRAYS
C
C POINTX(IDIR1) -  SENSOR CORDINATE OF A DETECTION OR MID-POINT PASSAGE
```

```
C                                                                       GRAP 560
C POINTY(IDIR1) -  TIME COORDINATE OF A DETECTION OR MID-POINT PASSAGE  GRAP 570
C                                                                       GRAP 580
C WESTLX(IDIR2) - | SEE                                                 GRAP 590
C WESTLY(IDIR2) - | THE                                                 GRAP 600
C WESTUX(IDIR2) - | EAST                                                GRAP 610
C WESTUY(IDIR2) - | DEFINITIONS                                         GRAP 620
C                                                                       GRAP 650
      SUBROUTINE GRAPH (ISWTCH, TIME,NPRNT)                             GRAP 660
      COMMON/GRAPHS/ DISTR , GRMIN , GRPMAX, GRPMIN, IDIM1 , IDIM2   ,
     1        KAGRAF, KEAST , KEASTR, KFALSE, KTRKS , KWEST , KWESTR  ,
     A       MDIM3 , MDIM4 , MEAST , MEASTR, MFALSE, MTRKS , MWEST ,
     B       MWESTR, NOWIND, NWQUIT, KEAST1, MEAST1, KWEST1, MWEST1,
     2        Z(200), AFALSX(200), AFALSY(200), DIRPEX( 25), DIRPEY( 25),
     3                DIRPWX( 25), DIRPWY( 25), EASTLX(150), EASTLY(150),
     4                EASTUX(150), EASTUY(150), POINTX(200), POINTY(200),
     5                WESTLX(150), WESTLY(150), WESTUX(150), WESTUY(150)
     6      ,TL(50),TAFLSX(100), TAFLSY(100), TDRPEX( 15), TDRPEY( 15),
     7              TDRPWX( 15), TDRPWY( 15), TESTLX( 75), TESTLY( 75),
     8              TESTUX( 75), TESTUY( 75), TPONTX(100), TPONTY(100),
     9              TWSTLX( 75), TWSTLY( 75), TWSTUX( 75), TWSTUY( 75)
C                                                                       GRAP 670
C**** IS IT FOR ITIAL GRAPH SETUP ONLY ?                                GRAP 680
C                                                                       GRAP 690
            IF (ISWTCH) 100, 100, 600                                   GRAP 700
C                                                                       GRAP 710
C**** NOT PLOT POINTS                                                   GRAP 720
C                                                 CHECK FOR ARRAY ERRORS GRAP 730
  100       IF (KWEST - IDIM2) 110, 110, 1000                           GRAP 740
  110       IF (KEAST - IDIM2) 120, 120, 1000                           GRAP 750
C                                                                       GRAP 760
C**** PLOT TRUCKS FIRST, USING + SIGN. (FOR ACTUAL DATA ALL DETECTIONS  GRAP 770
C                                           WILL BE IN THIS ARRAY)      GRAP 780
  120       IF (KTRKS) 140, 140, 130                                    GRAP 784
  130 CALL SETSMG (Z, 55, 0.0)                                          GRAP 790
      CALL SETSMG (Z, 53, .75)                                          GRAP 792
      CALL SETSMG (Z, 84, 1H+)                                          GRAP 800
      IF (NOWIND) 137, 134, 137                                         GRAP 802
  134 CALL SETSMG (Z, 55, 2.0 )                                         GRAP 804
      CALL SETSMG (Z, 84, 1H3)                                          GRAP 806
  137 CALL POINTG (Z, KTRKS, POINTY, POINTX)                            GRAP 810
               KTRKS = 0                                                GRAP 820
C                                                                       GRAP 840
C**** PLOT FALSE ALARMS                                                 GRAP 850
C                                                                       GRAP 860
  140       IF (KFALSE) 160, 160, 150                                   GRAP 870
  150 CALL SETSMG (Z, 55, 2.0)                                          GRAP 880
      CALL SETSMG (Z, 53, .75)                                          GRAP 882
      CALL SETSMG (Z, 84, 1H3)                                          GRAP 890
      CALL POINTG (Z, KFALSE, AFALSY, AFALSX)                           GRAP 900
               KFALSE = 0                                               GRAP 910
C**** PLOT WINDOWS                                                      GRAP 920
C                                               EAST FIRST - UPPER WIND GRAP 930
  160       IF (KEAST) 175, 175, 170                                    GRAP 940
  170 CALL SETSMG (Z, 55, 1.0)                                          GRAP 950
      CALL SETSMG (Z, 53, 1.5)                                          GRAP 952
      CALL SETSMG (Z, 84, 1HI)                                          GRAP 960
```

```
      CALL POINTG (Z, KEAST, EASTUY, EASTUX)                            GRAP 980
                 KEAST = 0                                              GRAP 982
C                                                   LOWER WINDOW        GRAP 990
  175     IF (KEAST1) 180,180,177                                       GRAP 994
  177 CALL SETSMG (Z, 84, 1H))                                          GRAP1000
      CALL POINTG (Z, KEAST1,EASTLY, EASTLX)                            GRAP1010
                 KEAST1 = 0                                             GRAP1020
C                                              NOW WEST - UPPER WINDOW  GRAP1030
  180       IF (KWEST) 195, 195, 190                                    GRAP1040
  190 CALL SETSMG (Z, 84, 1HL)                                          GRAP1050
      CALL SETSMG (Z, 53, 1.5)                                          GRAP1052
      CALL SETSMG (Z, 55, 2.0)                                          GRAP1060
      CALL POINTG (Z, KWEST, WESTUY, WESTUX)                            GRAP1062
                KWEST = 0                                               GRAP1072
  195         IF (KWEST1)  200, 200, 197                                GRAP1074
C                                                   LOWER WINDOW        GRAP1090
  197 CALL SETSMG (Z, 84, 1H>)                                          GRAP1100
      CALL SETSMG (Z, 55, 0.0)                                          GRAP1110
      CALL POINTG (Z, KWEST1,WESTLY, WESTLX)                            GRAP1120
                 KWEST1 = 0                                             GRAP1130
C                                                                       GRAP1140
C**** PLOT TRAJECTORY CONFIRMAION MARKS (<-)                             GRAP1150
C                                              WEST FIRST               GRAP1160
  200       IF (KWESTR) 220, 220, 210                                   GRAP1170
  210 CALL SETSMG(Z, 55, 2.0)                                           GRAP1180
      CALL SETSMG (Z, 53, 1.5)                                          GRAP1182
      CALL SETSMG(Z, 84, 1HZ)                                           GRAP1200
      CALL SETSMG (Z, 54, 90.)                                          GRAP1202
      CALL POINTG(Z, KWESTR, DIRPWY, DIRPWX)                            GRAP1210
      CALL SETSMG (Z, 54,  0.)                                          GRAP1212
                  KWESTR = 0                                            GRAP1220
C                                              EAST NEXT                GRAP1230
  220       IF (KEASTR) 300, 300, 230                                   GRAP1240
  230 CALL SETSMG(Z, 55, 2.0)                                           GRAP1250
      CALL SETSMG (Z, 53, 1.5)                                          GRAP1262
      CALL SETSMG(Z, 54, 1HY)                                           GRAP1270
      CALL SETSMG (Z, 54, 90.)                                          GRAP1282
      CALL POINTG(Z, KEASTR, DIRPEY, DIRPEX)                            GRAP1280
      CALL SETSMG (Z, 54,  0.)                                          GRAP1272
                  KEASTR = 0                                            GRAP1290
C                                                                       GRAP1300
C**** IF TIME EXCEEDS Y(MAX) PRINT GRAPH, ADVANCE TO NEXT GRAPH          GRAP1310
C                                                                       GRAP1320
C                                                                       GRAP1327
C**** TRANSFER OVERTIMES                                                GRAP1328
C                                                                       GRAP1329
  300 IF (GRPMAX + GRMIN - TIME)  320, 320, 310                         GRAP1330
  310 IF (NWQUIT) 320, 500, 320                                         GRAP1332
  320         NWQUIT = 0                                                GRAP1334
              GRPMIN = GRPMAY                                           GRAP1336
              GRPMAX = GRPMAX + GRMIN                                   GRAP1338
      CALL PAGEG (Z, 0, 1, 1)                                           GRAP1340
      CALL TRNSFR ( MEAST , KEAST , EASTUX, EASTUY, TESTUX, TESTUY)     GRAP1346
      CALL TRNSFR ( MEAST1, KEAST1, EASTLX, EASTLY, TESTLX, TESTLY)     GRAP1348
      CALL TRNSFR ( MWEST , KWEST , WESTUX, WESTUY, TWSTUX, TWSTUY)     GRAP1350
      CALL TRNSFR ( MWEST1, KWEST1, WESTLX, WESTLY, TWSTLX, TWSTLY)     GRAP1352
      CALL TRNSFR ( MTRKS , KTRKS , POINTX, POINTY, TPONTX, TPONTY)     GRAP1354
```

```
      CALL TRNSFR ( MFALSF, KFALSE, AFALSX, AFALSY, TAFLSX, TAFLSY)     GRAP1356
      CALL TRNSFR ( MEASTR, KEASTR, DIRPEX, DIRPEY, TORPEX, TORPEY)     GRAP1358
      CALL TRNSFR ( MWESTR, KWESTR, DIRPWX, DIRPWY, TORPWX, TORPWY)     GRAP1360
C                                             SETUP NEW GRAPH           GRAP1370
  400 CALL SUBJEG (Z, GRPMAX, 0., GRPMIN, 1.0)                          GRAP1380
      CALL OBJCTG (Z, .1, .1, 1.2333,  .9)                              GRAP1384
      CALL SETSMG (Z, 100, 3.0)                                         GRAP1388
      CALL GRIDG (Z, - 5.0, 0.0, 0, 0)                                  GRAP1392
      CALL SETSMG (Z, 14, 0.0)                                          GRAP1396
      CALL SETSMG (Z, 46, 90.)                                          GRAP1400
C                                                                       GRAP1404
            J = 0                                                       GRAP1408
      DO 410 I = 1, 11                                                  GRAP1412
            SPEC = GRPMAX - J *(GRPMAX - GRPMIN)/10.                    GRAP1416
            J = J + 1                                                   GRAP1420
  410 CALL NUMBRG (Z, SPEC, -.06, 4.0, SPEC)                            GRAP1424
            X =GRPMIN - GRMIN/100.                                      GRAP1428
      DO 420 I = 1, 50                                                  GRAP1432
      CALL NUMBRG (Z, X, TL(I), 2, I)                                   GRAP1436
            IF (TL(I) .GT. .949) GO TO 430                              GRAP1440
  420 CONTINUE                                                          GRAP1444
  430       X = GRPMIN - 3. * GRMIN/100.                                GRAP1448
      CALL LEGNDG (Z, X, .42, 13, 13HSENSOR NUMBER)                     GRAP1452
      CALL SETSMG (Z, 46, 0.)                                           GRAP1456
            X = GRPMAX - GRMIN * .4                                     GRAP1460
      CALL LEGNDG (Z, X,  -.07,11 ,11HTIME (MIN.))                      GRAP1464
  500 RETURN                                                            GRAP1470
  600 CALL MODESG (Z,0)                                                 GRAP1471
      GO TO 400                                                         GRAP1472
 1000 WRITE(NPRNT,1010)                                                 GRAP1472
 1010 FORMAT (49H0***** ERROR 117 : DECREASE THE NO. OF MIN./GRAPH )    GRAP1474
      CALL EXIT                                                         GRAP1476
      END                                                               GRAP1476
```

```
C ROUTINE READ                                  M. BERMAN  9/14/71
C
C     THIS ROUTINE READS THE TIME OF A DETECTION AND THE SENSOR NO. ON
C   WHICH IT OCCURS. FOR USE WITH THE CONVOY SIMULATOR IT READS WHEN A
C   CONVOY STARTS THRU THE STRING AND WHEN IT LEAVES. ALSO THE TIMES
C   A CONVOY BEGINS AND LEAVES AN OPERATING SENSORS SPHERE OF INFLUENCE.
C   THESE ADDITIONAL READS PERMIT KEEPING STATISTICS ON CONVOYS AND
C   TRUE TRAJECTORIES. WHEN THE PROGRAM IS USED SOLEY AS A LOGIC BOX
C   THIS ROUTINE SHOULD ONLY CONTAIN DETECTION READS.
C
      SUBROUTINE READ(BEGTME, FINTME, NRTRUK, NRCGEN, NSIZ3, NDISK,
     1                NRCNDT, NPRNT , ISENS , TIME  , NBRTRK,NTRAJC,
     2               LSTCNV,LASTSN,NWARAY, W,NB, NSENSR, NSNS         )
C
      COMMON/GRAPHS/ DISTR , GRMIN , GRPMAX, GRPMIN, IDIM1 , IDIM2   ,
     1       KAGRAF, KEAST , KEASTR, KFALSE, KTRKS , KWEST , KWESTR  ,
     A      MDIM3 , MDIM4 , MEAST , MEASTR, MFALSE, MTRKS , MWEST ,
     B      MWESTR, NOWIND, NWOUIT, KEAST1, MEAST1, KWEST1, MWEST1,
     2       Z(200), AFALSX(200), AFALSY(200), DIRPEX( 25), DIRPEY( 25),
     3               DIRPWX( 25), DIRPWY( 25), EASTLX(150), EASTLY(150),
     4               EASTUX(150), EASTUY(150), POINTX(200), POINTY(200),
     5               WESTLX(150), WESTLY(150), WESTUX(150), WESTUY(150)
     6     ,TL(50),TAFLSX(100), TAFLSY(100), TDRPEX( 15), TDRPEY( 15),
     7              TDRPWX( 15), TDRPWY( 15), TESTLX( 75), TESTLY( 75),
     8              TESTUX( 75), TESTUY( 75), TPONTX(100), TPONTY(100),
     9              TWSTLX( 75), TWSTLY( 75), TWSTUX( 75), TWSTUY( 75)
      DIMENSION BEGTME(NSIZ3,2), FINTME(NSIZ3, 2), NRTRUK(NSIZ3, 2),
     1          NRCGEN(1)       , NRCNDT(1)    ,LASTSN(1) ,LSTCNV(1), W(1)
C
C     EACH RECORD CONTAINS 5 WORDS:
C       WORD 1 - TIME
C       WORD 2 - (A)SENSOR NO. OR (B) NO. OF TRUCKS IN THE CONVOY
C       WORD 3 - CODE
C                1 - CONVOY STARTS THRU SENSOR FIELD
C                2 - CONVOY COMPLETES SENSOR FIELD
C                3 - FALSE ALARM DETECTION
C                4 - MID-POINT PASS (FIRST TRUCK)
C                5 - MID-POINT PASS (LAST TRUCK)
C                6 - TRUCK DETECTION
C                7 - FIRST TRUCK STARTS THRU A SENSOR
C                8 - LAST TRUCK LEAVES A SENSOR
C                9 - INDICATES END OF FILE
C       WORD 4 - CONVOY DIRECTION
C       WORD 5 - CONVOY NUMBER
C
   90 READ(NDISK) TIME, ISENS, ICODE, K, NRCNV
      GO TO (400, 500, 200, 300, 300, 100, 600, 700, 900), ICODE
C
C**** NORMAL TRUCK DETECTION
C
  100 IF (KAGRAF .EQ. 0 ) RETURN
C                                                GRAPH DETECTION
      IF (NWOUIT) 105, 103, 105
  103   IF (GRPMAX + GRMIN - TIME) 105, 105, 110
  105     ASSIGN 120 TO IRET
          GO TO 800
```

```
  110       IF (KTRKS - IDIM1) 120, 105, 105                                    READ 384
  120 IF (TIME .GE. GRPMAX) GO TO 125                                           READ 384
            KTRKS = KTRKS + 1                                                   READ 385
            POINTX(KTRKS) = TL(ISENS)                                           READ 386
            POINTY(KTRKS) = TIME                                                READ 387
            RETURN                                                              READ 388
  125       MTRKS = MTRKS + 1                                                   READ 389
            TPONTX(MTRKS) = TL(ISENS)                                           READ 390
            TPONTY(MTRKS) = TIME                                                READ 392
            IF (MTRKS .GE. MDIM3) NWQUIT = 5                                    READ 394
            RETURN                                                              READ 396
C                                                                               READ 400
C**** FALSE ALARM DETECTION                                                      READ 410
C                                                                               READ 420
  200 IF (KAGRAF .EQ. 0 ) RETURN                                                READ 428
C                                                    GRAPH DETECTION            READ 430
            IF (NWQUIT) 205, 203, 205                                           READ 431
  203       IF (GRPMAX + GRMIN - TIME) 205,205, 210                             READ 432
  205       ASSIGN 220 TO IRET                                                  READ 434
            GO TO 800                                                           READ 436
  210       IF (KFALSE - IDIM1) 220, 205, 205                                   READ 438
  220       IF (TIME .GE. GRPMAX) GO TO 225                                     READ 439
            KFALSE = KFALSE + 1                                                 READ 440
            AFALSX(KFALSE) = TL(ISENS)                                          READ 442
            AFALSY(KFALSE) = TIME                                               READ 444
            RETURN                                                              READ 446
  225       MFALSE = MFALSE + 1                                                 READ 447
             TAFLSX(MFALSE) = TL(ISENS)                                         READ 448
             TAFLSY(MFALSE) = TIME                                              READ 449
            IF (MFALSE .GE. MDIM3)NWQUIT = 6                                    READ 449
            RETURN                                                              READ 449
C                                                                               READ 450
C**** MID POINT PASS                                                            READ 460
C                                                                               READ 470
  300 IF((KAGRAF .EQ. 0) .OR. (NOWIND .EQ. 0)) GO TO 90                         READ 478
C                                                    GRAPH MID-POINT            READ 480
            IF (NWQUIT) 305, 303, 305                                           READ 481
  303       IF (GRPMAX + GRMIN - TIME) 305, 305, 310                            READ 482
  305       ASSIGN 320 TO IRET                                                  READ 484
            GO TO 800                                                           READ 486
  310       IF (KTRKS - IDIM1) 320, 305, 305                                    READ 488
  320       IF (K - 2*)  330, 340, 330                                          READ 491
  330              RSENSR = TL(ISENS) + .01                                     READ 492
                   GO TO 350                                                    READ 494
  340              RSENSR = TL(ISENS) - .01                                     READ 495
  350       IF (TIME .GE. GRPMAX) GO TO 370                                     READ 496
            KTRKS = KTRKS + 1                                                   READ 496
            POINTX(KTRKS) = RSENSR                                              READ 497
            POINTY(KTRKS) = TIME                                                READ 498
            GO TO 90                                                            READ 499
  370       MTRKS = MTRKS + 1                                                   READ 49A
            TPONTX(MTRKS) = RSENSR                                              READ 49B
            TPONTY(MTRKS) = TIME                                                READ 49C
            IF (MTRKS .GE. MDIM3) NWQUIT = 7                                    READ 49D
            GO TO 90                                                            READ 49E
C                                                                               READ 500
C***  CONVOY ENTERS THE STRING                                                   READ 510
```

```
C                                                                      READ 520
  400 WRITE(NPRNT,1) NRCNV, K, TIME, ISENS                             READ 530
C                                                 INCR. CNVS GENERATED READ 540
      NRCGEN(ISENS) = NRCGEN(ISENS) + 1                                READ 550
C                                                 NBR. TRUCKS IN CONV  READ 560
                N = MOD(NRCNV, NSIZ3)                                  READ 570
      IF (N) 1000, 410, 420                                            READ 580
  410           N = NSIZ3                                              READ 590
  420   NRTRUK(N,K) = ISENS                                            READ 600
      GO TO 90                                                         READ 630
C                                                                      READ 640
C**** CONVOY LEAVES STRING                                             READ 650
C                                                                      READ 660
  500 WRITE(NPRNT,1) NRCNV, K, TIME                                    READ 670
      GO TO 90                                                         READ 680
C                                                                      READ 690
C**** CONVOY STARTS THRU A SENSOR                                      READ 700
C                                             IS IT THE LAST SENSOR    READ 710
  600 IF (ISENS - LASTSN(K))90, 610, 90                                READ 720
C                                             YES. STORE BEGIN TIME    READ 730
  610         N = MOD(NRCNV, NSIZ3)                                    READ 740
      IF (N) 1000, 620, 630                                            READ 750
  620         N = NSIZ3                                                READ 760
  630 BEGTME(N,K) = TIME                                               READ 770
C                                              UPDTE LAST CNVY POINTER READ 772
         LSTCNV(K) = N                                                 READ 774
      GO TO 90                                                         READ 780
C                                                                      READ 790
C**** CONVOY COMPLETES A SENSOR                                        READ 800
C                                             IS IT THE LAST SENSOR    READ 810
  700 IF (ISENS - LASTSN(K)) 90, 710, 90                               READ 820
  710         N = MOD(NRCNV, NSIZ3)                                    READ 830
      IF (N) 1000, 720, 730                                            READ 840
  720         N = NSIZ3                                                READ 850
  730 FINTME(N,K) = TIME                                               READ 860
      GO TO 90                                                         READ 870
C                                                                      READ 871
C**** PLOT POINTS IF USER PERMITS                                      READ 872
C                                                                      READ 873
  800 CALL GRAPH  (0 , TIME, NPRNT)                                    READ 876
      GO TO IRET, (120, 220, 320)                                      READ 878
C                                                                      READ 880
C**** SIMULATION COMPLETE                                              READ 890
C                                                                      READ 900
  900 IF (KAGRAF) 910, 920, 910                                        READ 902
  910 IF (TIME .GT. GRPMAX) NWQUIT = 10                                READ 904
  915 CALL GRAPH (0,  TIME, NPRNT)                                     READ 906
      KADD = KEAST + KEASTR + KFALSE + KTRKS + KWEST + KWESTR          READ 907
     1     + KEAST1 + KWEST1                                           READ 907
      IF (KADD .GT. 0 ) GO TO 915                                      READ 907
      CALL EXITG(Z)                                                    READ 908
  920 CALL SUMRY(NRTRUK, NRCGEN, NSIZ3, NRCNOT, NPRNT, NBRTRK, NTRAJC, READ 910
     1             W     , NH    , NSENSR , NSNS  , NWARAY          )  READ 920
C                                                                      READ 930
C**** MESSAGES & FORMATS                                               READ 940
C                                                                      READ 950
 1000 WRITE(NPRNT, 1001) N, NRCNV, TIME                                READ 960
 1001 FORMAT(14H0** ERROR 300 , 2I10, F9.3)                            READ 970
      CALL EXIT                                                        READ 980
    1 FORMAT(1H , 80X, I6, 5X, I2, 3X, F9.3, 3X, I3 )                  READ 990
      END                                                              READ1000
```

```
C ... SUBROUTINE SETRAY                             23 NOV 1971           SRAY  10
C                                                                         SRAY  20
C   THIS ROUTINE IS USED TO SET THE ARRAYS FOR PRINTING WINDOWS.          SRAY  30
C                                                                         SRAY  40
      SUBROUTINE SETRAY (K, I, WUPTM, WLWTM )                             SRAY  50
      COMMON/GRAPHS/ DISTR , GRMIN , GRPMAX, GRPMIN, IDIM1 , IDIM2   ,
     1         KAGRAF, KEAST , KEASTR, KFALSE, KTRKS , KWEST , KWESTR  ,
     A        MDIM3 , MDIM4 , MEAST , MEASTR, MFALSE, MTRKS , MWEST ,
     B        MWESTR, NOWIND, NWQUIT, KEAST1, MEAST1, KWEST1, MWEST1,
     2         Z(200), AFALSX(200), AFALSY(200), DIRPEX( 25), DIRPEY( 25),
     3                 DIRPWX( 25), DIRPWY( 25), EASTLX(150), EASTLY(150),
     4                 EASTUX(150), EASTUY(150), POINTX(200), POINTY(200),
     5                 WESTLX(150), WESTLY(150), WESTUX(150), WESTUY(150)
     6      ,TL(50),TAFLSX(100), TAFLSY(100), TDRPEX( 15), TDRPEY( 15),
     7               TDRPWX( 15), TDRPWY( 15), TESTLX( 75), TESTLY( 75),
     8               TESTUX( 75), TESTUY( 75), TPONTX(100), TPONTY(100),
     9               TWSTLX( 75), TWSTLY( 75), TWSTUX( 75), TWSTUY( 75)
C                                                                         SRAY  70
      IF (K - 2) 20, 80, 20                                               SRAY  80
C                                                     WESTBOUND           SRAY  90
   20 IF (WUPTM .GE. GRPMAX) GO TO 30                                     SRAY 100
            KWEST = KWEST + 1                                             SRAY 110
      WESTUX(KWEST) = TL(I)                                               SRAY 120
      WESTUY(KWEST) = WUPTM                                               SRAY 130
   25 IF (WLWTM .GE. GRPMAX) GO TO 40                                     SRAY 140
            KWEST1 = KWEST1 + 1                                           SRAY 150
      WESTLX(KWEST1) = TL(I)                                              SRAY 160
      WESTLY(KWEST1) = WLWTM                                              SRAY 170
            RETURN                                                        SRAY 180
   30       MWEST = MWEST + 1                                             SRAY 190
      TWSTUX(MWEST) = TL(I)                                               SRAY 200
      TWSTUY(MWEST) = WUPTM                                               SRAY 210
      IF (MWEST .GE. (MDIM4 - 5)) NWQUIT=1                                SRAY 212
            GO TO 25                                                      SRAY 220
   40       MWEST1 = MWEST1 + 1                                           SRAY 230
      TWSTLX(MWEST1) = TL(I)                                              SRAY 240
      TWSTLY(MWEST1) = WLWTM                                              SRAY 250
      IF (MWEST1 .GE. (MDIM4 - 5)) NWQUIT=2                               SRAY 252
             RETURN                                                       SRAY 260
C                                                     EASTBOUND           SRAY 270
   80 IF (WUPTM .GE. GRPMAX) GO TO 90                                     SRAY 280
            KEAST = KEAST + 1                                             SRAY 290
      EASTUX(KEAST) = TL(I)                                               SRAY 300
      EASTUY(KEAST) = WUPTM                                               SRAY 310
   85 IF (WLWTM .GE. GRPMAX) GO TO 100                                    SRAY 320
            KEAST1 = KEAST1 + 1                                           SRAY 330
      EASTLX(KEAST1) = TL(I)                                              SRAY 340
      EASTLY(KEAST1) = WLWTM                                              SRAY 350
            RETURN                                                        SRAY 360
   90       MEAST = MEAST + 1                                             SRAY 370
      TESTUX(MEAST) = TL(I)                                               SRAY 380
      TESTUY(MEAST) = WUPTM                                               SRAY 390
      IF (MEAST .GE. (MDIM4 - 5)) NWQUIT=3                                SRAY 392
             GO TO 85                                                     SRAY 400
  100       MEAST1 = MEAST1 + 1                                           SRAY 410
      TESTLX(MEAST1)= TL(I)                                               SRAY 420
      TESTLY(MEAST1)= WLWTM                                               SRAY 430
      IF (MEAST1 .GE. (MDIM4 - 5)) NWQUIT=4                               SRAY 432
             RETURN                                                       SRAY 440
      END                                                                 SRAY 450
```

```
C   SUMRY ROUTINE                     M. BERMAN 9/27/71                        SUMY  10
C                                                                              SUMY  20
      SUBROUTINE SUMRY (NRTRUK, NRCGEN, NSIZ3 , NRCNDT, NPRNT , NBRTRK,         SUMY  30
     1           NTRAJC,  W      , NB     , NSENSR, NSNS  ,                    SUMY  32
     2                    NWARAY                                              )SUMY  33
C                                                                              SUMY  40
      COMMON/XTRA/ TIMUNO(50), TIMFIN(50), WPRIME(15,150), FCTSEN, RHO
     1              , NA(100) ,JVALMN(100), JASTMN(100)
      DIMENSION NRTRUK(NSIZ3,2), NRCGEN(1), NRCNDT(1), W(NSNS,NWARAY) ,        SUMY  50
     1             SUMX(50),         SUMXX(50), AVGN(50) , STD(50)             SUMY  60
C                                                                              SUMY  70
C**** DETECTION STATISTICS                                                     SUMY  80
C                                                                              SUMY  90
      WRITE(NPRNT,1)                                                           SUMY 100
            ITOT1 = 0                                                          SUMY 110
            ITOT2 = 0                                                          SUMY 120
      DO 20  I = 1, NBRTRK                                                     SUMY 130
   10               ITOT1 = ITOT1 + NRCGEN(I)                                  SUMY 150
                    ITOT2 = ITOT2 + NRCNDT(I)                                  SUMY 160
   20 WRITE(NPRNT,2) I, NRCGEN(I), NRCNDT(I)                                   SUMY 170
C                                                     TOTALS                   SUMY 180
   30 WRITE(NPRNT,3) ITOT1, ITOT2                                              SUMY 190
C                                                                              SUMY 200
            IPHANT = NTRAJC - ITOT2                                            SUMY 210
      WRITE(NPRNT,4) IPHANT, NTRAJC                                            SUMY 220
            ISET = 1                                                           SUMY 234
            GO TO 34                                                           SUMY 234
C                                                                              SUMY 230
C**** PRINT WEIGHT ARRAY AND GET MEAN AND VARIANCE FOR EACH SENSOR              SUMY 240
C                                                                              SUMY 250
   32       ISET = 0                                                           SUMR 252
            WRITE(NPRNT,9)                                                     SUMY 253
   34 DO 35 I = 1, NSENSR                                                      SUMY 254
                 SUMX(I) = 0.0                                                 SUMY 256
   35           SUMXX(I) = 0.0                                                 SUMY 258
                 NT = 1                                                        SUMY 260
                 NP  = 15                                                      SUMY 270
   40 IF (NP .GT. NSENSR) NP = NSENSR                                          SUMY 280
      WRITE(NPRNT,5) (I, I = NT, NP )                                          SUMY 290
            JDIM = NTRAJC/NB                                                   SUMY 300
      DO 60 J = 1, JDIM                                                        SUMY 310
      WRITE(NPRNT,6)J,(W(I,J), I = NT , NP )                                   SUMY 320
      DO 60 I = NT, NP                                                         SUMY 330
            SUMX(I) = SUMX(I) + W(I,J)                                         SUMY 340
          SUMXX(I) = SUMXX(I) + W(I,J) * W(I,J)                                SUMY 350
   60       W(I,J) = WPRIME(I,J)                                               SUMY 355
C                                              CALCULATE MEAN AND VAR          SUMY 360
      DO 70 I = NT, NP                                                         SUMY 370
            AVGN(I) = SUMX(I)/JDIM                                             SUMY 380
   70       STD(I) =((( JDIM*SUMXX(I))- (SUMX(I)*SUMX(I)))/                    SUMY 390
     1                                  (JDIM*(JDIM-1))) ** .5                 SUMY 400
      WRITE(NPRNT, 7)  (AVGN(K), K = NT, NP    )                               SUMY 410
      WRITE(NPRNT, 8) (STD(L),  L = NT, NP)                                    SUMY 414
C                                                                              SUMY 420
      IF (  NP  .GE. NSENSR) GO TO 80                                          SUMY 430
               NT  = NP + 1                                                    SUMY 440
```

```
            NP = NP + 15                                              SUMY 450
         GO TO 40                                                     SUMY 460
   80 IF (ISET .EQ. 1) GO TO 32                                       SUMY 464
      WRITE(NPRNT,11)                                                 SUMY 465
C                          TEMPORARY FOR PRINTING AVERAGE ASTI'S & VALID
      WRITE(NPRNT,13)
      DO 92 I = 1, NSENSR
      AVGAST = FLOAT(JASTMN(I))/FLOAT(JDIM)
      AVGVAL = FLOAT(JVALMN(I))/FLOAT(JDIM)
   92 WRITE(NPRNT,14) I, AVGVAL, AVGAST
      CALL EXIT                                                       SUMY 468
  100 RETURN                                                          SUMY 470
    1 FORMAT(1H1, 20X, 32H*** CONVOY DETECTION SUMMARY *** // 15X,    SUMY 480
     1           11HCONVOY SIZE, 4X, 13HNO. GENERATED, 4X ,           SUMY 490
     2           12HNO. DETECTED//)                                   SUMY 500
    2 FORMAT(20X, I3, 12X, I4, 13X, I4)                               SUMY 510
    3 FORMAT(12X,  8HTOTALS :, 14X, I5, 12X, I5)                      SUMY 520
    4 FORMAT(1H0, 12X,15HPHANTOM TRAJ = , I3,19H TOTAL CONFIRMED = , I6)SUMY 530
    5 FORMAT(1H1,53X, 37H*** SENSOR WEIGHTS BY TIME PERIOD ***//      SUMY 540
     1         12H TIME PERIOD,  56X, 6HSENSOR/ 15X, 15(I2,5X)//)     SUMY 550
    6 FORMAT( 5X,I3, 4X, 15(F5.3,2X))                                 SUMY 560
    7 FORMAT( 8HOMEAN : , 4X, 15(F6.4,1X))                            SUMY 570
    8 FORMAT(11HOSTD DEV : ,1X, 15(F6.4,1X))                          SUMY 580
    9 FORMAT (1H0, 20X, 32H(THESE WEIGHTS ARE NOT SMOOTHED))          SUMY 584
   11 FORMAT (1H0, 20X, 28H(THESE WEIGHTS ARE SMOOTHED) )             SUMY 584
   13 FORMAT(1H1,'SENSOR', 2X,'AVG.VALIDS', 2X, 'AVG.ASTIS'//)        SUMY 585
   14 FORMAT( 3X,I3,3X,F10.3,2X,F10.3)                                SUMY 586
      END                                                             SUMY 590
```

```
C ROUTINE TRAJCM                               M. BERMAN   9/15/71
C
C     THIS ROUTINE IS CALLED WHEN A CONJECTURED TRAJECTORY IS CLOSED TO
C   SEE IF IT IS CONFIRMED. IF IT IS THEN WE CHECK TO SEE IF ENOUGH
C   CONFIRMATIONS HAVE BEEN MADE TO CALCULATE NEW WEIGHTS (A NEW TIME
C   PERIOD IS STARTED). WHEN WEIGHTS ARE COMPUTED ANY HYPO-ACTIVE
C   SENSOR IS REMOVED FROM THE STRING.
C
      SUBROUTINE TRAJCM(NRTJ  , NRCEL , K     , TIME  , IDROP , CSENS ,
     1                  INACTV, LASTSN, NASTC, NB ,ND, NDRSTR, NTJSTR,
     2                  MSENS , NV    , NTRAJC, R     , TSBAR , WCAP  ,
     3              W, NSNS   , NSIZ3 , NRTJC ,NBRTRK, BEGTME, FINTME,
     4                  NCNDT , NRTRUK, NPRNT ,LSTCNV, NSN   , NSENSR,F,
     5                  TVEL  , NWARAY                                  )
C
C     NRTJ - TRAJECTORY NUMBER
C     NRCEL- TRAJECTORY CELL NO.
C     TIME - TIME OF TRAJECTORY CLOSURE
C
C
      COMMON/XTRA/ TIMUNO(50), TIMFIN(50), WPRIME(15,150), PCTSEN, RHO
     1           , NA(100) ,JVALMN(100), JASTMN(100)
      COMMON/GRAPHS/ DISTR , GRMIN , GRPMAX, GRPMIN, IDIM1 , IDIM2   ,
     1       KAGRAF, KEAST , KEASTR, KFALSE, KTRKS , KWEST , KWESTR  ,
     A      MDIM3 , MDIM4 , MEAST , MEASTR, MFALSE, MTRKS , MWEST ,
     B      MWESTR, NOWIND, NHQUIT, KEAST1, MEAST1, KWEST1, MWEST1,
     2       Z(200), AFALSX(200), AFALSY(200), DIRPEX( 25), DIRPEY( 25),
     3                DIRPWX( 25), DIRPWY( 25), EASTLX(150), EASTLY(150),
     4                EASTUX(150), EASTUY(150), POINTX(200), POINTY(200),
     5                WESTLX(150), WESTLY(150), WESTUX(150), WESTUY(150),
     6      TL(50),TAFLSX(100), TAFLSY(100), TDRPEX( 15), TDRPEY( 15),
     7              TDRPWX( 15), TDRPWY( 15), TESTLX( 75), TESTLY( 75),
     8              TESTUX( 75), TESTUY( 75), TPONTX(100), TPONTY(100),
     9              TWSTLX( 75), TWSTLY( 75), TWSTUX( 75), TWSTUY( 75)
      DIMENSION  NASTC(NRTJC, 2, NSNS),         NTJSTR(1), NDRSTR(1),
     1           LASTSN(1)    ,           NV(1), R(1), INACTV(1), IDROP(1)
     2         , TSBAR(NSN,2), BEGTME(1), FINTME(1), NCNDT(1),
     3           NRTRUK(1), LSTCNV(1)    , TVEL(NSN,2,2), W(NSNS,NWARAY)
C
            NDRIND = 0
               NAS = 0
                AW = 0.0
                N2 = (NTRAJC + NB)/NB
C
C**** COUNT AND WEIGHT THE ADMISSABLE STRIPS IN THE TRAJECTORY
C
      DO 100  I = 1, NSENSR
C                                                   ADMSBLE STRIPS
              NAS = NAS + NASTC(NRTJ,K,I)
C                                                   WEIGHT OF STRIPS
  100           AW = AW + NASTC(NRTJ,K,I) * WPRIME(I,N2)
C
C
C**** DO WEIGHTS AND NO. OF STRIPS CONFIRM A TRAJECTORY ?
C
      IF (NAS - MSENS) 120, 110, 110
  110 IF (AW - WCAP) 120, 200, 200
```

```
  120 RETURN                                                          TRAJ 410
C                                                                     TRAJ 420
C**** TRAJECTORY CONFIRMED                                            TRAJ 430
C                                                                     TRAJ 440
  200 NTRAJC = NTRAJC + 1                                             TRAJ 442
          N = MOD(NTRAJC,NB)                                          TRAJ 443
      IF (N) 220, 205, 220                                            TRAJ 443
  205      N = NB                                                     TRAJ 445
C                                        SUM ADM STRIPS CONTRIBUTED   TRAJ 460
  220 DO 210 I = 1 , NSENSR                                           TRAJ 470
  210 NA(I) = NA(I) + NASTC(NRTJ,K,I)                                 TRAJ 480
      WRITE(NPRNT, 1) TIME, K, LASTSN(K)                              TRAJ 530
C                                                                     TRAJ 531
C**** SET GRAPH OUTPUT                                                 TRAJ 532
C                                                                     TRAJ 533
      IF ( (KAGRAF .EQ. 0) .OR. (NOWIND .EQ. 0) ) GO TO 280           TRAJ 533
            IF (K - 2)  225, 230, 225                                 TRAJ 534
  225       IF (TIME .GE. GRPMAX) GO TO 227                           TRAJ 535
                  KWESTR = KWESTR + 1                                 TRAJ 536
      DIRPWX(KWESTR) = .975                                           TRAJ 537
      DIRPWY(KWESTR) =  TIME                                          TRAJ 538
                  GO TO 280                                           TRAJ 539
  227             MWESTR = MWESTR + 1                                 TRAJ 53A
      TORPWX(MWESTR) = .975                                           TRAJ 53B
      TORPWY(MWESTR) =  TIME                                          TRAJ 53C
                  GO TO 280                                           TRAJ 53D
  230       IF (TIME .GE. GRPMAX) GO TO 232                           TRAJ 53E
                  KEASTR = KEASTR + 1                                 TRAJ 53F
      DIRPEX(KEASTR) = .025                                           TRAJ 53G
      DIRPEY(KEASTR) = TIME                                           TRAJ 53H
                  GO TO 280                                           TRAJ 53I
  232             MEASTR = MEASTR + 1                                 TRAJ 53J
      TORPEX(MEASTR) = .025                                           TRAJ 53K
      TORPEY(MEASTR) = TIME                                           TRAJ 53L
  280 CALL CNTTRJ(TIME  , F     , K     , BEGTME, FINTME, NRTRUK, NCNDT,TRAJ 540
     1            NSIZ3 , LSTCNV                                  )   TRAJ 550
C                                                                     TRAJ 560
C**** HAVE B TRAJECTORIES BEEN CONFIRMED ?                             TRAJ 570
C                                                                     TRAJ 580
      IF (N - NB) 120, 300, 300                                       TRAJ 590
C                                                                     TRAJ 600
C**** YES! TIME PERIOD IS COMPLETED. COMPUTE NEW WEIGHTS.              TRAJ 610
C                                                                     TRAJ 620
  300         SUMR = 0.0                                              TRAJ 630
                 N2 = N2 + 1                                          TRAJ 634
      IF (N2 .GT. NWARAY) GO TO 1080                                  TRAJ 636
      WRITE(NPRNT,5) (NV(J), J=1,NSENSR)                              TRAJ 637
      WRITE(NPRNT,6) (NA(J), J=1,NSENSR)                              TRAJ 638
  310 DO 390 I = 1, NSENSR                                            TRAJ 640
C                                         RATIO ADM STRPS TO VALIDTRAJ 720
             IF (NV(I)) 370, 360, 370                                 TRAJ 722
  360           R(I) = 0.0                                            TRAJ 724
                GO TO 380                                             TRAJ 726
  370 R(I) = AMAX1(0.,FLOAT(NA(I))*(AMIN1(1.,(1.+((FLOAT(NA(I))/      TRAJ 730
     1FLOAT(NB))**2)-(FLOAT(NV(I))/FLOAT(4*NB))))))                    
C                                         ZERO VALID STRIP CNTR.      TRAJ 740
  380      JVALMN(I) = JVALMN(I) + NV(I)                              TRAJ 742
```

```
           JASTMN(I) = JASTMN(I) + NA(I)
              NV(I) = 0
              NA(I) = 0
  390           SUMR = SUMR + R(I)
C
C**** USING R & SUM OF R CALCULATE WEIGHTS FOR THE NEW TIME PERIOD
C
      DO 500  I = 1, NSENSR
         W(I,N2) = R(I)/SUMR
      WPRIME(I,N2) = (1 -RHO) * W(I,N2) + RHO * WPRIME(I,(N2-1))
C                                             IS WT. BELOW MINIMUM'S ?
      IF (IDROP(I)) 1030, 398, 500
  398 IF (WPRIME(I,N2) - CSENS) 400, 400, 490
C                                             YES! INCR. CNTR.
  400    INACTV(I) = INACTV(I) + 1
C                                             MORE THAN D PERIODS ?
      IF (INACTV(I) - ND) 500, 410, 500
C                                             YES! REMOVE SENSOR I
  410     IDROP(I) = 1
      WRITE(NPRNT,3) I
            NDRIND = 1
      GO TO 500
C                                             NO! ZERO INACTIVE CNTR.
  490    INACTV(I) = 0
  500 CONTINUE
      GO TO 550
  550 WRITE(NPRNT,2) ( W(I,N2), I= 1, NSENSR)
      WRITE(NPRNT,4) (WPRIME(I,N2) , I = 1,NSENSR)
C
C**** IF A SENSOR WAS DROPPED UPDATE THE SENSOR STRING
C
      IF (NDRIND) 120, 120, 600
  600       LIVSNS = 1
                KL = LASTSN(1)
  605           NL = LASTSN(2)
C                                         (A) DO WESTBOUND UPDT FIRST
      IF (IDROP(NL)) 1030, 700, 610
C                                             THE FIRST SENSOR IS GONE
  610    LASTSN(2) = NL + 1
      GO TO 605
C                                             CHECK MID SENSORS
  700           NL = NL + 1
      IF (IDROP(NL)) 1030, 730, 710
C                                             UPDATE TIME BTWN SENSORS
  710 TSBAR(NL+1,1) = TSBAR(NL+1,1) + TSBAR(NL,1)
      TVEL(NL+1,1,1) = TVEL(NL+1,1,1) + TVEL(NL,1,1)
      TVEL(NL+1,1,2) = TVEL(NL+1,1,2) + TVEL(NL,1,2)
      TVEL(NL,1,1) = 0.0
      TVEL(NL,1,2) = 0.0
       TSBAR(NL,1) = 0.0
      GO TO 740
  730    LASTSN(1) = NL
            LIVSNS = LIVSNS + 1
C                                             SEE IF THRU THE STRING
  740 IF (NL - KL) 700, 750, 750
C                                             QUIT IF TOO FEW SENSORS
  750 MSENS = PCTSEN * LIVSNS
```

```
      IF (MSENS - 2)       1070,  800, 800                                  TRAJ1190
C                                              (B) DO EASTBOUND DIRECTION   TRAJ1200
  800            NL = LASTSN(1)                                             TRAJ1210
                 KL = LASTSN(2)                                             TRAJ1220
  805            NL = NL - 1                                                TRAJ1230
      IF (IDROP(NL)) 1030, 830, 810                                         TRAJ1240
C                                                 UPDATE TIME BTWN SENSORSTRAJ1250
  810 TSBAR(NL-1, 2) = TSBAR(NL-1,2) + TSBAR(NL,2)                          TRAJ1260
      TVEL(NL-1,2,1) = TVEL(NL-1,2,1) + TVEL(NL,2,1)                        TRAJ1262
      TVEL(NL-1,2,2) = TVEL(NL-1,2,2) + TVEL(NL,2,2)                        TRAJ1264
      TVEL(NL,2,2) = 0.0                                                    TRAJ1268
      TVEL(NL,2,1) = 0.0                                                    TRAJ1266
        TSBAR(NL,2)  = 0.0                                                  TRAJ1270
  830 IF (NL - KL) 120, 120, 805                                            TRAJ1280
C                                                                           TRAJ1290
C**** MESSAGES AND FORMATS                                                  TRAJ1300
C                                                                           TRAJ1310
 1030 WRITE(NPRNT,1031) IDROP(NL), NL , KL                                  TRAJ1350
 1031 FORMAT(15H0*** ERROR 410 , 3I10)                                      TRAJ1360
      CALL EXIT                                                             TRAJ1370
 1070 WRITE(NPRNT,1071) MSENS, LIVSNS, (IDROP(I), I = 1, NSENSR)            TRAJ1380
 1071 FORMAT(21H0THERE ARE LESS THAN , I3, 23H SENSORS IN THE STRING.,      TRAJ1390
     1      11H THERE ARE , I3/1H0,(40(I2,1X))    )                         TRAJ1400
      CALL EXIT                                                             TRAJ1410
 1080 WRITE(NPRNT, 1081) N2, NWAFAY                                         TRAJ1412
 1081 FORMAT(15H0*** ERROR 413 , 2I6)                                       TRAJ1413
      CALL EXIT                                                             TRAJ1414
    1 FORMAT(20H0TRAJ. CONFIRMED AT , F8.3, 6H DIR. , I2, 8H SENSOR ,I3)TRAJ1420
    2 FORMAT( 19H THE UPDATED W(I) :,2X, 15F6.3/(21X,15F6.3))               TRAJ1422
    3 FORMAT(28H0REMOVED HYPOACTIVE SENSOR :, I3)                           TRAJ1427
    4 FORMAT( 19H THE    WPRIM (I) :,2X, 15F6.3/(21X,15F6.3))               TRAJ1428
    5 FORMAT(7H VALIDS,2X ,20I4/ (9X,20I4))                                 TRAJ1429
    6 FORMAT(7H ASTI'S,2X ,20I4/ (9X,20I4))                                 TRAJ1429
      END                                                                   TRAJ1430
```

```
C     SUBROUTINE TRANSFER                                     23 NOV 71
C
C     TRANSFER THE OVERTIMES TO THE PRIMARY GRAPHING ARRAYS
C
C     M1 - THE NUMBER OF OVERTIMES IN THE ARRAY
C     M2 - THE COUNTER OF THE PRIMARY GRAPHING ARRAY
C   E(I) - THE EAST GRAPHING ARRAY
C   W(I) - THE WEST GRAPHING ARRAY
C  TE(I) - THE EAST OVERTIME ARRAY
C  TW(I) - THE WEST OVERTIME ARRAY
C
      SUBROUTINE TRNSFR (M1, M2, E, W, TE, TW)
C
      DIMENSION E(1), W(1), TE(1), TW(1)
C
      IF (M1) 10, 100, 10
   10       DO 20  I = 1, M1
                M2 = M2 + 1
              E(M2) = TE(I)
   20         W(M2) = TW(I)
                M1 = 0
C
  100 RETURN
      END
```

```
C  ROUTINE VALIDS                              M. BERMAN  10 SEPT 71
C
C     THIS ROUTINE CHECKS EACH DETECTION AS IT ARRIVES TO SEE IF IT
C    SHOULD BE ADDED TO A VALID STRIP. IF NOT IT CHECKS TO SEE IF ANY
C    CURRENT OPEN WINDOWS SHOULD BE CLOSED.  IF A VALID STRIP IS
C    COMPLETED IT IS CHECKED FOR ADMISSABILITY AFTER WINDOWS ON ALL OTHER
C    SENSORS ARE CHECKED FOR CLOSURE.
C
      SUBROUTINE VALIDS(IDROP, IKTRAJ, INACTV, LASTSN, BWWND, NAVLID,
     1                  NASTC, NDRSTR, NDETEC, NOPENT, NRTRAJ, NTJSTR,
     2                  NV, R, TMEFST, TMELST, TSBAR , UPWND , W      ,
     3                  WLWTM, WUPTM , IB    , BEGTME, FINTME, NRGEN ,
     4                  NCNDT, NRTRUK, NRTJC , NBRTRK, NEND  , NSIZ3 ,
     5                  NSNS , BETA  , IWCNT , NSENSR, MSENS , CSENS ,
     6                  ND   , NB    , NTRAJC, LSTCNV , NDISK , NPRNT ,
     7                  JSR    , TMESR, WCAP ,NWARAY                   )
C
      DIMENSION  IDROP(1), NDETEC(1), TMEFST(1), TMELST(1), NV(1)  ,
     1           IKTRAJ(1), INACTV(1), LASTSN(1), BWWND(1), NAVLID(1),
     2           NASTC(1), NDRSTR(1), NOPENT(1), NRTRAJ(1), NTJSTR(1),
     3           R(1), TSBAR(1), UPWND(1), W(1), WLWTM(1), WUPTM(1),
     4           IB(1), BEGTME(1), FINTME(1), NRGEN(1),NCNDT(1),
     5           NRTRUK(1), LSTCNV(1), JSR(1)    ,TMESR(1)
        KSWTCH = 0
       JSWTCH = 0
C
C**** READ THE SENSOR NO. AND TIME OF DETECTION FROM THE DISK
C
  100 CALL READ(BEGTME, FINTME, NRTRUK, NRGEN , NSIZ3, NDISK, NCNDT ,
     1          NPRNT, I , TIME, NBRTRK, NTRAJC, LSTCNV, LASTSN        ,
     2          NWARAY, W    , NB   , NSENSR, NSNS                     )
       MSWTCH = 0
C
C**** IS THIS DETECTION THE BEGINING OF A NEW STRIP ?
C
      TMECHK = TIME - TMELST(I) - BETA
      IF (TMECHK) 105, 105, 110
C
C**** NO! CONTINUE THE STRIP - UPDATE LAST DETECTION COUNTER
C
  105        TMELST(I) = TIME
             NDETEC(I) = NDETEC(I) + 1
             GO TO 100
C
C**** YES! ITS THE BEGINING OF A NEW STRIP. WAS THE PREVIOUS ONE VALID?
C
  110 IF (NDETEC(I) - IWCNT) 112, 116, 116
  112       JSWTCH = 0
            MSWTCH = 1
            GO TO 209
C
C**** STRIP WAS NOT VALID. CHECK ALL WINDOWS TO SEE IF ANY SHOULD CLOSE.
C
  115 CALL CHKWIN(IDROP, IKTRAJ, INACTV, LASTSN, BWWND, NAVLID, NASTC ,
     1           NDRSTR, NPRNT , NOPENT, NRTRAJ, NTJSTR,NV, R ,TMEFST(I)
     2        ,TMELST(I),TSBAR , UPWND , W      , WLWTM ,WUPTM , IB    ,
     3           BEGTME, FINTME, NRGEN , NCNDT , NRTRUK,NRTJC , NBRTRK,
```

```
     4           NEND   , NSI23 , NSNS   , BETA   , IWCNT ,NSENSR, MSENS , VALD 440
     5           CSENS , ND, NB, I, 0   , NTRAJC, LSTCNV, WCAP ,THELST,  VALD 450
     5           TMEFST, NWARAY                                       ) VALD 454
      GO TO 130                                                         VALD 460
C                                                                       VALD 470
C**** YES ITS A VALID STRIP. SEE IF ANY WINDOWS ON OTHER SENSORS SHOULD VALD 480
C**** BE CLOSED. INCREMENT VALID STRIP COUNTER.                         VALD 490
C                                                                       VALD 500
  116        NV(I) = NV(I) + 1                                          VALD 510
C                                                                       VALD 511
C**** CHEECK ALL OTHER SENSORS FOR VALID STRIPS                         VALD 512
C                                                                       VALD 513
  209             J = 0                                                 VALD 514
              ISORT = 0                                                 VALD 515
             ASSIGN 123 TO MASSN                                        VALD 516
  117      .      J = J + 1                                             VALD 517
           IF (J - NSENSR) 300, 300, 204                                VALD 518
  300      IF ( J- I) 118, 117, 118                                     VALD 519
  118      IF (NDETEC(J) - IWCNT) 117, 301, 301                         VALD 51A
  301      IF ( TIME - TMELST(J) - BETA) 117, 117, 119                  VALD 51A
C                                                                       VALD 51B
C**** STRIP IS VALID - PLACE IT IN LIST                                 VALD 51C
C                                                                       VALD 51D
  119         ISORT = ISORT + 1                                         VALD 51E
      TMESR(ISORT) = TMELST(J)                                          VALD 51F
        JSR(ISORT) = J                                                  VALD 51G
             NV(J) = NV(J) + 1                                          VALD 51G
       NDETEC(J) = 0                                                    VALD 51G
      IF (TMELST(J) .GE. TMELST(I)) JSWTCH = 1                          VALD 51G
             KSWTCH = 1                                                 VALD 51H
             GO TO 117                                                  VALD 51H
C                                                   SORT LIST ON LOW TIME  VALD 51I
  120            NN1 = NSENSR - 1                                       VALD 51J
             DO 122  K1 = 1, NN1                                        VALD 51K
                K2 = K1 + 1                                             VALD 51L
             DO 122  K3 = K2, NSENSR                                    VALD 51M
          IF (TMESR(K1) - TMESR(K3)) 122, 122, 121                      VALD 51N
  121         TEMP = TMESR(K1)                                          VALD 51O
          TMESR(K1) = TMESR(K3)                                         VALD 51P
          TMESR(K3) = TEMP                                              VALD 51Q
              ITEMP = JSR(K1)                                           VALD 51R
            JSR(K1) = JSR(K3)                                           VALD 51S
            JSR(K3) = ITEMP                                             VALD 51T
  122        CONTINUE                                                   VALD 51T
C                                                                       VALD 51V
C**** FOR EACH VALID STRIP CLOSE ALL WINDOWS AND CHECK FOR ADMISIBILITY.VALD 51W
C                                                                       VALD 51X
              ISORT = 0                                                 VALD 51Y
  123         ISORT = ISORT + 1                                         VALD 51Z
                 J = JSR(ISORT)                                         VALD 520
             IF (J - 99999) 124, 201, 201                               VALD 521
C .                                            RESET SENSOR SORT LIST   VALD 522
  201        DO 203  K1 = 1, ISORT                                      VALD 523
          TMESR(K1) = 999999.                                           VALD 524
  203      JSR(K1) = 99999                                              VALD 524
            KSWTCH = 0                                                  VALD 524
  204   IF (KSWTCH) 205, 205, 120                                       VALD 524
```

```
  205 IF (MSWTCH) 206, 207, 206                                           VALD 524
  206 IF (JSWTCH) 115, 115, 130                                           VALD 524
  207   ASSIGN 130 TO MASSN                                               VALD 526
                J = I                                                     VALD 527
  124 CALL CHKWIN(IDROP , IKTRAJ, INACTV, LASTSN, BWWND , NAVLID, NASTC,VALD 529
     1            NDRSTR, NPRNT , NOPENT, NRTRAJ, NTJSTR, NV,R,TMEFST(J)VALD 530
     2       ,TMELST(J) , TSBAR , UPWND , W     , WLWTM , WUPTM , IB   , VALD 540
     3            BEGTME, FINTME, NRGEN , NCNDT , NRTRUK, NRTJC , NBRTRKVALD 550
     4          , NEND  , NSIZ3 , NSNS  , BETA  , IWCNT , NSENSR, MSENS,VALD 560
     5            CSENS , ND    , NB    , J     , 1     , NTRAJC,LSTCNV,VALD 570
     6            WCAP  , TMELST, TMEFST, NWARAY                      ) VALD 570
C                                                                         VALD 580
C**** CHECK THIS VALID STRIP FOR ADMISSABILITY IF SENSOR IS IN STRING.   VALD 590
C                                                                         VALD 600
      IF (IDROP(J)) 1000, 125, 127                                        VALD 610
  125 CALL CHKADM(IDROP , IKTRAJ, INACTV, LASTSN, BWWND , NAVLID, NASTC VALD 620
     1          , NDRSTR, NPRNT , NOPENT, NRTRAJ, NTJSTR, NV,R,TMEFST(J)VALD 630
     2        ,TMELST(J), TSBAR , UPWND , W     , WLWTM , WUPTM , IB    ,VALD 640
     3            BEGTME, FINTME, NRGEN , NCNDT , NRTRUK, NRTJC , NBRTRKVALD 650
     4          , NEND  , NSIZ3 , NSNS  , BETA  , IWCNT , NSENSR, MSENS,VALD 660
     5            CSENS , ND    , NB    , J     , NTRAJC ,LSTCNV, WCAP, VALD 670
     6            NWARAY                                               ) VALD 670
  127 GO TO MASSN, (123, 130)                                             VALD 675
C                                                                         VALD 680
C**** START A NEW STRIP                                                   VALD 690
C                                                                         VALD 700
  130         TMEFST(I) = TIME                                            VALD 710
              TMELST(I) = TIME                                            VALD 720
              NDETEC(I) =  1                                              VALD 730
      GO TO 100                                                           VALD 740
C                                                                         VALD 750
C**** ERROR MESSAGES                                                      VALD 760
C                                                                         VALD 770
 1000 WRITE(NPRNT, 1)  IDROP(J), J, TIME                                  VALD 780
    1 FORMAT(15H0**** ERROR 200, 2I10, F10.3)                             VALD 790
      RETURN                                                              VALD 800
      END                                                                 VALD 810
```